IPG Politécnico da Guarda
Polytechnic of Guarda

RELATÓRIO DE PROJETO

Licenciatura em Engenharia Informática

Pedro Daniel Tomás de Almeida

dezembro | 2015

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

RELATÓRIO DE PROJETO

GESTÃO DE PRESENÇAS

PEDRO DANIEL TOMÁS DE ALMEIDA

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO EM ENGENHARIA INFORMÁTICA

Novembro/2015

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

RELATÓRIO DE PROJETO

GESTÃO DE PRESENÇAS

PEDRO DANIEL TOMÁS DE ALMEIDA

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO EM ENGENHARIA INFORMÁTICA

Novembro/2015

ORIENTADOR: NOEL DE JESUS MENDONÇA LOPES

Elementos Identificativos

Aluno

Nome: Pedro Daniel Tomás de Almeida
Número: 1009263
Curso: Engenharia Informática

Estabelecimento de Ensino

Escola Superior de Tecnologia e Gestão - Instituto Politécnico da Guarda
Morada: Av. Dr. Francisco Sá Carneiro 50, 6300-559 Guarda
Telefone: 271220120/271220146 | Fax: 271220150

Duração do Projeto

Inicio: 18 de Maio de 2015
Fim: 30 de Novembro de 2015

Orientador do Projeto

Nome: Noel de Jesus Mendonça Lopes
Grau académico: Doutor

Resumo

Este documento descreve o projeto desenvolvido no âmbito da Unidade Curricular Projeto de Informática, na licenciatura em Engenharia Informática na Escola Superior de Tecnologia e Gestão do Instituto Politécnico da Guarda.

Cada vez mais a assiduidade é um fator muito importante para os professores, pois muitas vezes é determinante nas notas do aluno. Tendo em conta que o tradicional método da assinatura na folha de papel é facilmente falsificável, procurou-se obter uma solução para este problema.

Este projeto tem como objetivo facilitar o controlo de assiduidade por parte dos professores e também dos alunos, através de uma aplicação web. Neste documento são descritas todas as fases de desenvolvimento deste projeto.

Palavras-Chave: Web, ASP.NET, Base de Dados, Assiduidade, Ensino

Abstract

This document describes the project developed under the discipline Computer Project, on degree in Computer Engineering at the School of Technology and Management of the Polytechnic Institute of Guarda.

More and more attendance is a very important factor for teachers, as it is often determinant in student grades. Given that the traditional method signature on the paper sheet is easily falsifiable, we search to get a solution to this problem.

This project aims to facilitate the attendance control by teachers as well as students, through a web application. This document describes all the stages of development of this project.

Keywords: Web, ASP.NET, Database, Attendance, Education

Agradecimentos

Em primeiro lugar, quero agradecer á minha família e amigos por todo o apoio e incentivo dado ao longo da realização do projeto.

Agradeço também ao meu orientador de projeto Professor Noel de Jesus Mendonça Lopes, pelo seu profissionalismo, dedicação e disponibilidade demonstrada ao longo do projeto.

Índice

Índice de Figuras

Índice de Tabelas

1. Introdução

Este relatório tem o objetivo de descrever o projeto desenvolvido pelo aluno Pedro Daniel Tomás de Almeida, no âmbito da Unidade Curricular Projeto de Informática, na licenciatura em Engenharia Informática na Escola Superior de Tecnologia e Gestão do Instituto Politécnico da Guarda.

Gerir as presenças dos alunos nas aulas sempre foi algo importante para os professores para poderem ter uma noção correta da assiduidade dos alunos, que muitas vezes é determinante na classificação final dos mesmos. Com a tecnologia presente no nosso dia-a-dia nas mais diversas áreas já não faz sentido os professores fazerem a gestão da assiduidade dos alunos pelo tradicional método da assinatura na folha de presenças, que muitas vezes é falsificável.

Deste modo procurei desenvolver uma solução para este problema, nomeadamente uma aplicação web onde os alunos podem marcar a sua presença. O professor posteriormente confirma essa mesma presença ou não. É também possível a marcação de aulas e configuração de limites de tolerância, sendo que ultrapassado o limite o aluno já não consegue marcar a presença. O professor e o aluno têm também acesso a relatórios da assiduidade.

O projeto consistiu na análise dos requisitos necessários para o desenvolvimento do mesmo, desenho de protótipos, e desenvolvimento da solução proposta, tendo sido realizados durante o desenvolvimento e depois do desenvolvimento vários testes ao funcionamento da aplicação.

1.1. Motivação

Um dos motivos que me levaram a realização deste projeto foi a oportunidade de desenvolver as minhas capacidades no desenvolvimento de aplicações web, mais propriamente na tecnologia da Microsoft ASP.NET.

Outro dos aspetos foi a realização do projeto de um modo mais profissional, pois sendo realizado com a orientação do meu orientador de projeto, o projeto ganha um teor mais sério do que um projeto académico para uma outra unidade curricular, permitindo assim desenvolver competências e métodos mais profissionais para o futuro.

Em suma, foi possível aprofundar os meus conhecimentos nas linguagens C#, HTML e CSS, o que para mim era importante, pois programação para a web é uma das minhas áreas de interesse e muitas empresas atualmente procuram programadores nesta área.

1.2. Objetivos

Uma das variáveis que mais afeta a obtenção da aprovação em unidades curriculares é assiduidade dos alunos. Cada vez mais há a necessidade dos professores saberem se os alunos assistem realmente às aulas ou não, pois o tradicional método da assinatura na folha de presenças é facilmente falsificável.

Este projeto tem como objetivo a implementação de uma aplicação web com o intuito de gerir as presenças dos alunos nas aulas. Este projeto terá como público-alvo os alunos do ensino superior e deverá permitir:

- O aluno deve registar a sua presença através da plataforma.
- No fim da aula o professor deverá confirmar a presença dos alunos que estiveram na aula.
- Configurar limites de tolerância.
- Validação de justificação de faltas.
- Criação de turmas.
- Criação de relatórios com a folha de presenças.

A tecnologia a utilizar para o desenvolvimento do projeto é ASP.net, e a metodologia a usar é o Desenvolvimento Ágil.

1.3. Estrutura do Documento

Este documento é composto por mais cinco capítulos para além do presente capítulo, estando organizado da seguinte forma:

- O segundo capítulo incide sobre o estudo e levantamento do estado da arte: são apresentadas aplicações já existentes no mercado similares à aplicação desenvolvida.

- No terceiro capítulo é descrita a metodologia e tecnologias utilizadas ao longo do desenvolvimento deste projeto, bem como a calendarização das tarefas executadas.

- No quarto capítulo é descrita a análise dos requisitos necessários ao desenvolvimento da aplicação.

- O quinto capítulo é a descrição da implementação da aplicação proposta com imagens de resultados da aplicação desenvolvida.

- No sexto e último capítulo são apresentadas as conclusões finais acerca do projeto, tais como eventuais melhorias futuras.

2. Estado da arte

Este Capítulo irá abordar um estudo feito às aplicações existentes nesta área da gestão da assiduidade. As aplicações estudadas são aplicações que embora realizem o objetivo principal proposto no desenvolvimento desta solução, não se focam unicamente nesse aspeto. Sendo que uma delas é configurável para gerir também funcionários por exemplo. As restantes são especialmente focadas no percurso académico dos alunos, mais propriamente nas classificações dos alunos e nas comunicações entre os pais dos mesmos.

2.1. Aplicações existentes

Das várias aplicações nesta área existentes no mercado selecionei quatro para análise, MyAT (Figura 1), GradeLink (Figura 2), MyGradeBook (Figura 3) e o plugin Autoattendance block.

2.1.1. MyAttendanceTracker

A MyAt é uma aplicação online gratuita de registo de presenças e possui um conjunto de funcionalidades, como demonstra a Figura 1, que permite para além da gestão de presenças, a criação de turmas, gestão das notas dos alunos, criação de relatórios relativos à situação do aluno na aula, como por exemplo as suas presenças ou notas. Uma das principais características é a comunicação entre professores/pais havendo também alertas por parte da aplicação caso as notas do aluno comecem a baixar ou caso o aluno falte as aulas. A MyAt permite também:

- Gestão dos alunos;
- Relatórios standards ou relatórios personalizáveis com informações dos alunos;
- Categorias de gestão de presenças personalizável;
- Ecrãs com toda a informação acerca dos alunos, turmas e notas;
- Aplicação otimizada para dispositivos móveis;

- Gestão de utilizadores, permitindo a criação de utilizadores para os pais poderem consultar a informação dos seus educandos;

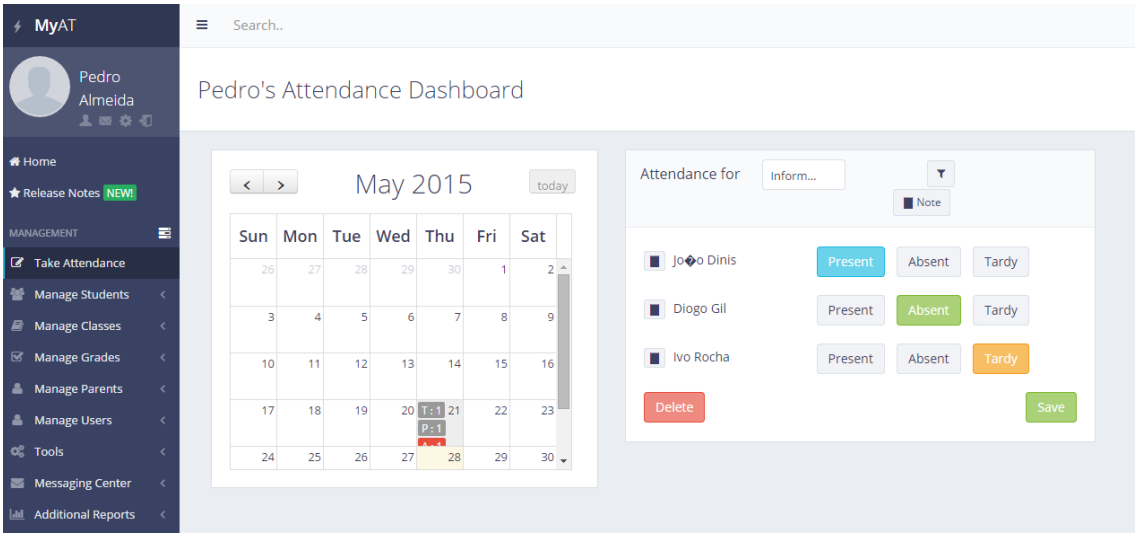

Figura 1- Ecrã da aplicação MyAttendanceTracker

2.1.2. Gradelink

A Gradelink é uma aplicação online direcionada não só para os professores mas também para os alunos e pais, como se pode ver na Figura 2, envolve custos e reúne a informação relativa aos alunos. É possível gerir a presença dos alunos nas aulas bem como as suas notas às respetivas disciplinas. As disciplinas podem também ser inseridas/editadas. Permite pagamentos referentes a despesas escolares tais como almoços ou a compra de um livro escolar. Outras das características da Gradelink são:

- Gestão de alunos;
- Gestão de turmas e horários, permitindo a inserção de turmas e visualização do respetivo horário;
- Relatórios personalizáveis com informações dos alunos;
- Comunicação com os pais, enviando alerta acerca das notas ou presenças;
- Registos médicos dos alunos;

Figura 2 – Ecrã da aplicação Gradelink

2.1.3. MyGradeBook

A MyGradeBook é uma aplicação online que permite a publicação segura de notas de alunos do ensino básico e secundário desde 1999. Para além da publicação de notas permite também a gestão de presenças dos alunos, como se pode verificar na Figura 3, impressão de relatórios, envio de emails aos pais dos alunos. A MyGradeBook é uma aplicação que envolve custos mas no entanto permite uma demonstração grátis de trinta dias. A MyGradeBook permite também:

- Opcionalmente permite o login de pais e alunos para consultar a informação;
- Relatórios personalizáveis;
- Criação de turmas;
- Criação de testes online;
- Criação de contas de grupo;

Figura 3 – Ecrã da aplicação MyGradeBook

2.1.4. Autoattendance block

O Autoattendance block é um plugin para a plataforma e-learning Moodle. Para a utilização deste plugin é necessário ter instalado o módulo Attendance na plataforma. O Autoattendance block permite aos professores um acesso rápido à funcionalidade que permite gerir as presenças dos alunos e permite aos alunos de igual modo acesso aos relatórios de presenças na disciplina. O plugin pode funcionar em três modos distintos: o modo automático em que as presenças são registadas assim que o aluno efetua login na disciplina na plataforma Moodle, neste modo é possível a utilização de restrições por endereços IP; o modo semi-automático em que os alunos marcam a sua presença na plataforma mas são os professores a validar a sua presença, neste modo é possível também a utilização de restrições por endereços IP ou através de uma palavra-chave definida; o modo manual em que os professores fazem a tradicional chamada e marcam eles próprios a presença dos alunos. Para além das já referidas o plugin Autoattendance block possui ainda as seguintes características:

- Gestão de turmas;
- Criação de relatórios de assiduidade;

2.2. Análise Critica

Das aplicações estudadas todas elas reúnem o objetivo principal deste projeto, a gestão de presenças dos alunos nas aulas, sendo que apenas a MyAttendanceTracker é gratuita. A MyAttendanceTracker tem uma particularidade que as outras aplicações estudadas não têm, sendo ela a possibilidade de personalizar a categoria de gestão de presenças, ou seja a aplicação pode ser personalizada para gerir presenças de alunos ou presenças de funcionários. A Gradelink e a MyGradeBook são apenas direcionadas para o ensino, mais propriamente o ensino básico e secundário, a MyAttendanceTracker não tem um público-alvo específico pois tem a particularidade de ser personalizável para alunos ou funcionários. Uma característica que distingue a Gradelink das outras aplicações estudadas e a possibilidade de realização de pagamentos de despesas escolares através da aplicação. O plugin Autoattendance Block tem a funcionalidade de gestão de presenças mas em comparação com as outras aplicações estudadas no que diz respeito á gestão dos alunos ou da disciplina é mais incompleto, quanto ao público-alvo, como o Autoattendance Block é um plugin para a plataforma e-learning Moodle, pode abranger para além do ensino básico e secundário também o ensino superior. Em Suma todas as aplicações estudadas são muito parecidas no que diz respeito aos objetivos principais diferindo apenas em algumas características como demonstra a tabela seguinte:

	MyAttendanceTracker	**Gradelink**	**MyGradeBook**	**Autoattendance Block**
Gestão de Presenças	Sim	Sim	Sim	Sim
Gestão de Notas	Sim	Sim	Sim	Não
Gestão de Turmas	Sim	Sim	Sim	Sim
Relatórios	Sim	Sim	Sim	Sim
Realização de pagamentos	Não	Sim	Não	Não
Custos	Não	Sim	Sim	Não
Plataforma e-learning	Não	Não	Não	Sim
Público-alvo	Estudantes/Funcionários	Estudantes	Estudantes	Estudantes

Tabela 1- Comparação das aplicações estudadas

3. Metodologia e Resultados

Este capítulo vai falar da metodologia usada no desenvolvimento deste projeto, o planeamento e descrição das tarefas a realizar, e os resultados desse planeamento.

3.1. Metodologia

No desenvolvimento de software é importante termos sempre a opinião do cliente final e tendo em conta que o desenvolvimento de software passa por diversas etapas torna-se ainda mais importante o feedback do cliente à medida que o projeto evolui. Normalmente as necessidades do cliente ou a especificação do produto final vão-se alterando à medida que o projeto avança sendo por isso necessária uma cooperação constante com o cliente (Grando, 2015). Assim sendo a metodologia a ser utilizada no desenvolvimento deste projeto será o desenvolvimento ágil, mais especificamente o desenvolvimento ágil XP, pois permite o desenvolvimento de software de forma iterativa e incremental com capacidade para reagir ao feedback do cliente final. Tem como princípios principais:

- A satisfação do cliente é a prioridade. Há total cooperação entre a equipa que desenvolve o projeto e o cliente.
- Software funcional em vez de documentação.
- Resposta às modificações mesmo estando em etapas avançadas.

As etapas de desenvolvimento que este projeto deverá seguir são as seguintes:

- Análise dos requisitos do projeto
- Implementação da aplicação
- Realização de testes
- Documentação do projeto num relatório

Para o desenvolvimento do projeto será usada a plataforma da Microsoft ASP.NET, utilizando as linguagens de programação C#, HTML e CSS. A base de dados que servirá de suporte

à aplicação irá utilizar o SGBD da Microsoft, o SQL Server. O desenvolvimento do layout da aplicação irá ser feito com a ajuda do Bootstrap, uma coleção de ferramentas gratuitas e open-source para a criação de aplicações web que contêm componentes como botões, formulários e outros componentes de interface baseados em HTML e CSS, facilitando assim o desenvolvimento de aplicações web como e o caso deste projeto (Wikipedia, 2015).

3.2. Descrição das tarefas

As tarefas principais são:

- Tarefa 1 – Análise dos requisitos do projeto;
- Tarefa 2 – Implementação da aplicação;
- Tarefa 3 – Realização de testes;
- Tarefa 4 – Documentação do projeto num relatório;

Tarefa	Nome da Tarefa	Duração	Início	Fim
1	Análise dos requisitos do projeto	75 dias	18-05-2015	31-07-2015
2	Implementação da aplicação	61 dias	01-08-2015	30-09-2015
3	Realização de testes	15 dias	01-10-2015	15-10-2015
4	Documentação do projeto num relatório	16 dias	16-10-2015	31-10-2015

Tabela 2 – Tarefas Previstas

O respetivo mapa de gantt é apresentado na figura seguinte:

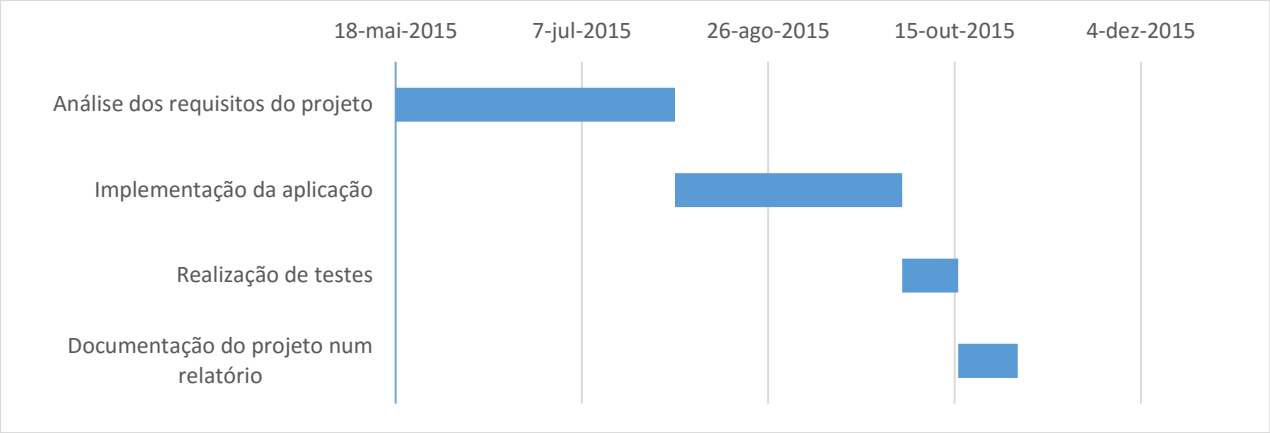

Figura 4 – Mapa de Gantt previsto

Nem tudo o que é descrito neste Mapa de Gantt foi cumprido com sucesso no tempo previsto sendo que o mapa de Gantt cumprido é mostrado na tabela e na figura seguinte:

Tarefa	Nome da Tarefa	Duração	Início	Fim
1	Análise dos requisitos do projeto	75 dias	18-05-2015	31-07-2015
2	Implementação da aplicação	102 dias	01-08-2015	30-09-2015
3	Realização de testes	5 dias	01-10-2015	15-10-2015
4	Documentação do projeto num relatório	15 dias	16-10-2015	31-10-2015

Tabela 3 – Tarefas Cumpridas

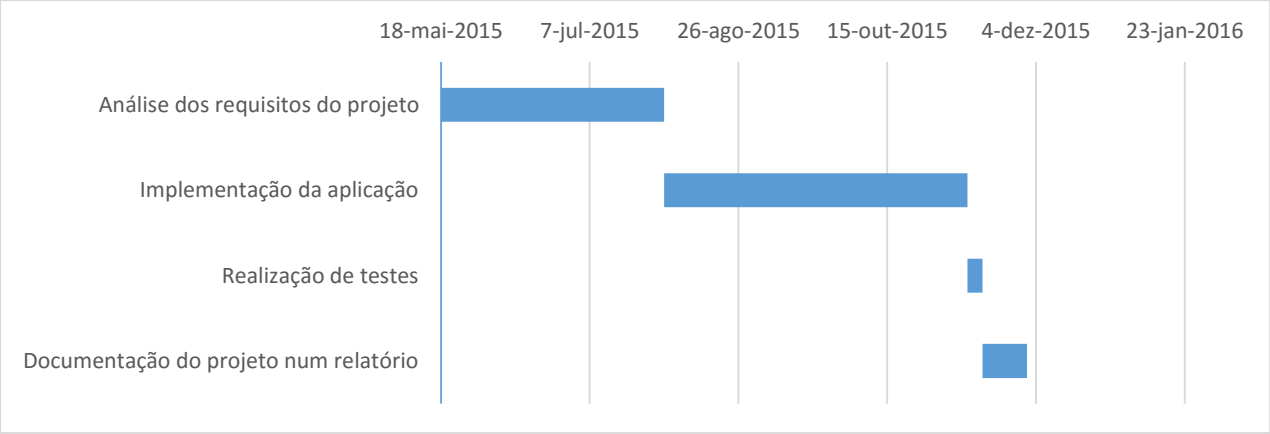

Figura 5 – Mapa de Gantt Cumprido

3.3. Resultados da Análise

Depois do estudo e planeamento das tarefas para o desenvolvimento deste projeto espera-se:

- Aplicação web responsiva;
- Facilidade nos alunos em marcar a sua assiduidade;
- Facilidade nos professores em verificar a assiduidade dos alunos;
- Facilidade no administrador em gerir as contas dos alunos, professores e o tempo de tolerância pretendido para a marcação da assiduidade;
- Maior controlo da assiduidade por parte do aluno e também do professor.

4. Análise dos Requisitos

Neste capítulo é feito uma análise dos requisitos necessários para o desenvolvimento deste projeto. Para esta análise foi utilizada a linguagem UML, que é uma linguagem utilizada pelos programadores para poderem visualizar os seus projetos na forma de diagramas (Wikipédia, UML, 2015).

4.1. Diagrama de Contexto

O diagrama de contexto é composto por fluxos de dados que mostram as interfaces entre o sistema e entidades externas, sendo assim uma forma de representar o projeto e a sua relação ao ambiente (Wikipédia, Diagrama de contexto, 2015). O diagrama de contexto descreve assim a ideia geral do projeto facilitando assim a sua compreensão. Na figura 6 está representado o diagrama de contexto elaborado para este projeto.

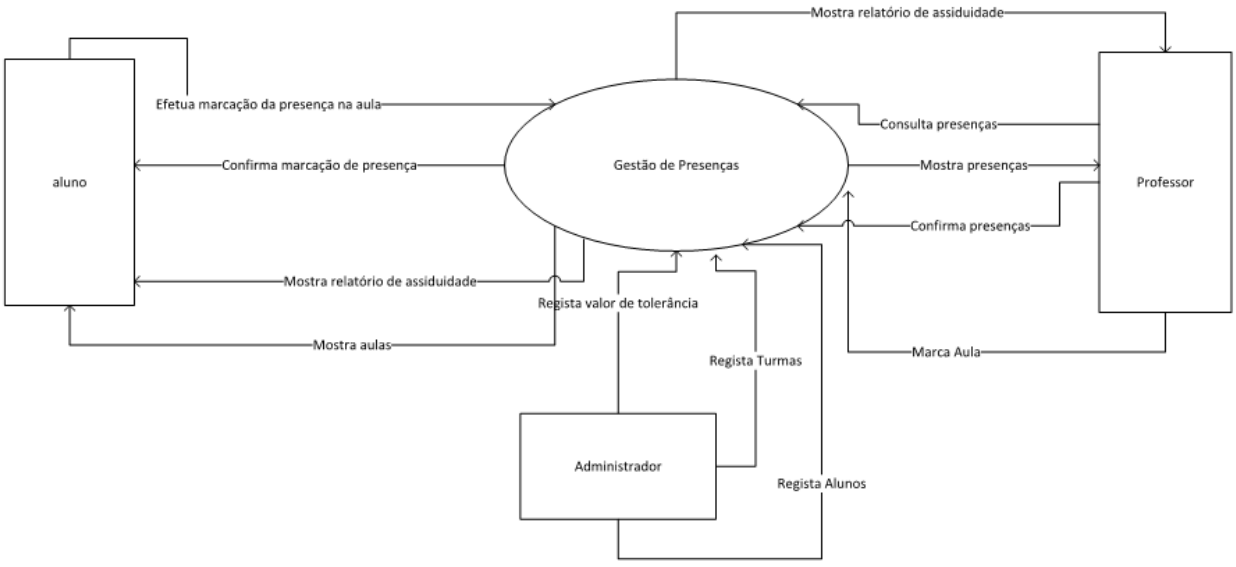

Figura 6 –Diagrama de Contexto

4.1.1. Descrição do diagrama de contexto

- O aluno efetua a marcação da assiduidade que é guardada no sistema, onde posteriormente o professor vai validar ou não a assiduidade;
- O aluno efetua uma consulta da sua assiduidade;
- O sistema mostra ao aluno a sua assiduidade;
- O sistema mostra a lista de assiduidade ao professor;
- O professor valida a assiduidade;
- O professor efetua a marcação de uma aula;
- O professor efetua uma consulta da assiduidade dos alunos;
- O sistema mostra ao professor a consulta;
- O administrador regista o valor da tolerância permitida para a marcação da assiduidade;
- O administrador regista conta de aluno;
- O administrador regista turmas;

4.2. Casos de Uso

Os diagramas de casos de uso são uma excelente ferramenta para o levantamento de requisitos para o desenvolvimento de um projeto pois descreve as funcionalidades propostas para o projeto (Wikipédia, Diagrama de caso de uso, 2015). Permite definir o ator e a sua interação com o sistema. Na figura 7 está representado o diagrama de casos de uso para este projeto.

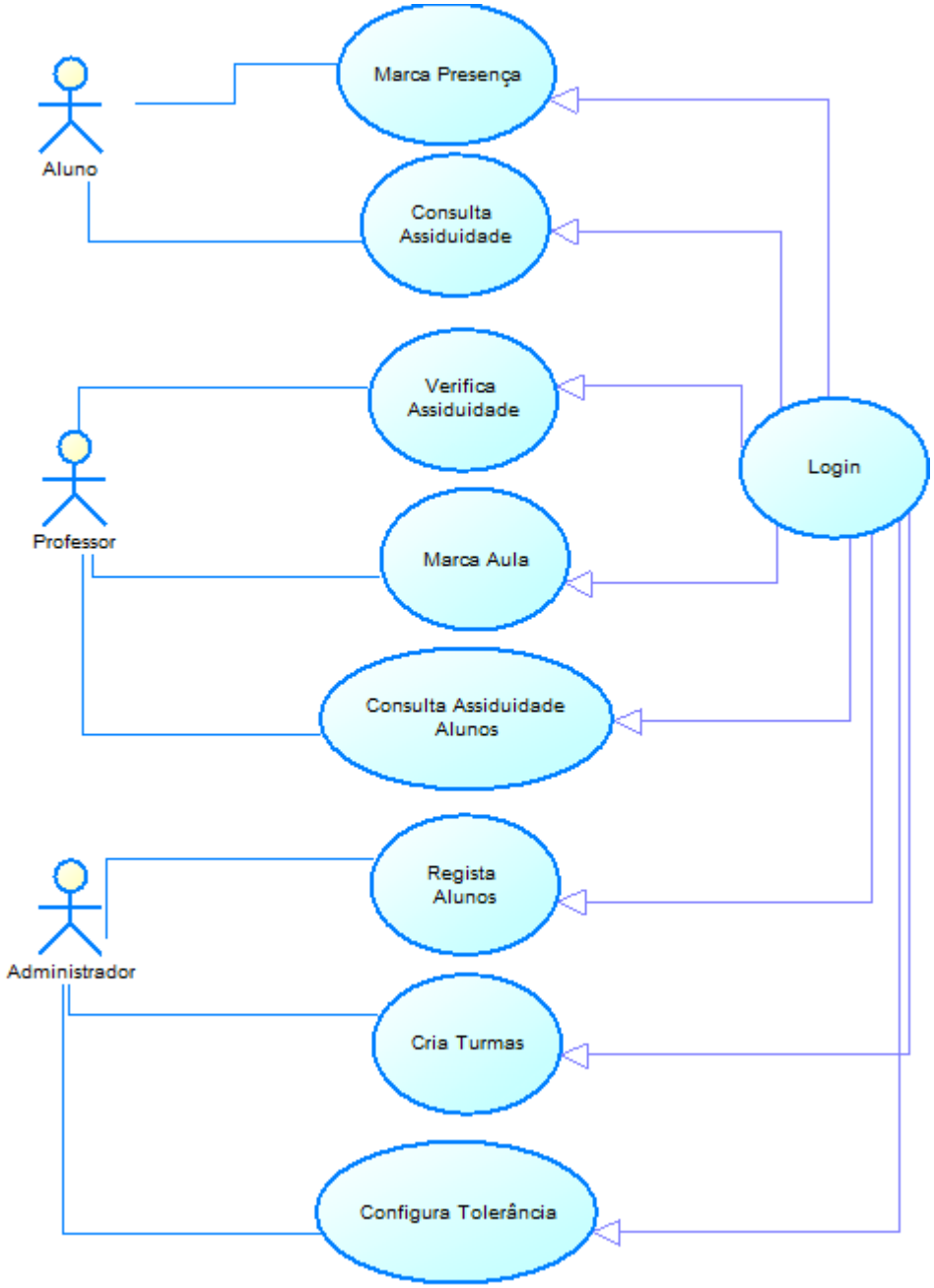

Figura 7 – Diagrama de casos de uso

4.2.1. Descrição dos casos de uso

Os casos de uso necessários para a realização deste projeto são descritos em baixo e são os seguintes: Marcar Presença (Tabela 4), Consulta Assiduidade (Tabela 6), Verifica Assiduidade (Tabela 7), Marcar Aula (Tabela 9), Consulta Assiduidade Alunos (Tabela 11), Regista Alunos (Tabela 12), Cria Turmas (Tabela 14), Configura Tolerância (Tabela 16).

Caso de uso Marcar Presença

Nome	**Marcar Presença**
Atores	Aluno
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal efetuar a marcação da assiduidade do aluno
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Aluno escolhe a opção presenças 2. Sistema mostra as aulas disponíveis do aluno 3. Aluno efetua a marcação da assiduidade 4. O sistema guarda a assiduidade do aluno
Caminhos alternativos	2. a) Não tem aulas marcadas para esse dia 3. a) Está fora do limite para a marcação da assiduidade 4. a) Aluno cancela
Suplementos ou Adornos	Testar numa aula que já passou
Pós-condições	O sistema guarda a assiduidade para posteriormente o professor aprovar ou não.

Tabela 4 – Caso de uso Marcar Presença

Caso de teste - Marcar Presença

Caso de Teste:	**Marcar Presença**
Objetivo:	Verificar se é possível marcar presença numa aula que já passou
Inputs	Nome e password do utilizador
Procedimentos:	1. Introduzir dados de utilizador 2. Selecionar aula 3. Verificar se e possível introduzir no sistema
Outputs	Dados da assiduidade

Tabela 5 – Caso de teste Marcar Presença

Caso de uso Consulta Assiduidade

Nome	Consulta Assiduidade
Atores	Aluno
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal efetuar a consulta de assiduidade do aluno
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Aluno escolhe a opção ver assiduidade 2. Sistema mostra a assiduidade do aluno
Caminhos alternativos	2. a) O aluno não tem assiduidade
Suplementos ou Adornos	
Pós-condições	

Tabela 6 – Caso de uso Consulta Assiduidade

Caso de uso Verifica Assiduidade

Nome	Verifica Assiduidade
Atores	Professor
Prioridade	Alta (Crítica)
Descrição	Este caso de uso tem como objetivo principal efetuar a validação da assiduidade do aluno.
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Professor escolhe a opção verificar assiduidade. 2. Sistema mostra a assiduidade dos alunos. 3. Professor seleciona aluno e aprova ou não a assiduidade. 4. O sistema guarda a assiduidade.
Caminhos alternativos	2. a) Não há assiduidade. 4. a) O professor cancela.
Suplementos ou Adornos	Testar se assiduidade ficou realmente aprovada ou não
Pós-condições	O sistema guarda a assiduidade para mais tarde ser consultada

Tabela 7 – Verifica Assiduidade

Caso de teste – Verifica Assiduidade

Caso de Teste:	**Verifica Assiduidade**
Objetivo:	Verificar se a assiduidade ficou realmente aprovada
Inputs	Nome e password do utilizador
Procedimentos:	1. Introduzir dados de utilizador 2. Selecionar disciplina 3. Verificar se a assiduidade realmente sofreu alterações ou não.
Outputs	Dados da assiduidade

Tabela 8 – Caso de teste Verifica Assiduidade

Caso de uso Marcar Aula

Nome	**Marcar Aula**
Atores	Professor
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal efetuar a marcação de uma aula.
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Professor escolhe a opção marcar aula. 2. Sistema mostra as aulas disponíveis. 3. Professor introduz os dados da nova aula. 4. O sistema guarda a aula.
Caminhos alternativos	2. a) Não há aulas. 3. a) Dados não válidos. 4. a) Professor cancela.
Suplementos ou Adornos	Testar se a aula ficou marcada
Pós-condições	O sistema guarda a aula para depois ser apresentada aos alunos para ser efetuada a sua assiduidade.

Tabela 9 – Marcar Aula

Caso de teste – Marcar Aula

Caso de Teste:	**Marcar Aula**
Objetivo:	Verificar se a aula ficou marcada e visível para os alunos
Inputs	Nome e password do utilizador, dados da aula
Procedimentos:	1. Introduzir dados de utilizador 2. Selecionar a opção Marcar Aula 3. Introduzir dados da aula 4. Verificar se a aula ficou marcada
Outputs	Dados da aula

Tabela 10 – Caso de teste Marcar Aula

Caso de uso Consulta Assiduidade Alunos

Nome	**Consulta Assiduidade Alunos**
Atores	Professor
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal efetuar a consulta da assiduidade dos alunos.
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Professor escolhe a opção Relatórios. 2. Sistema mostra as opções disponíveis. 3. Professor seleciona as opções. 4. O sistema mostra assiduidade.
Caminhos alternativos	4. a) Não há assiduidade.
Suplementos ou Adornos	
Pós-condições	

Tabela 11 – Consulta Assiduidade Alunos

Caso de uso Regista Alunos

Nome	**Regista Alunos**
Atores	Administrador
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal efetuar a introdução de um novo aluno no sistema.
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Administrador escolhe a opção alunos. 2. Administrador introduz os dados do aluno. 3. O sistema guarda dados do aluno.
Caminhos alternativos	2. a) Dados não válidos. 3. a) O administrador cancela.
Suplementos ou Adornos	Testar se os dados são corretamente inseridos (ex. campos obrigatórios todos preenchidos).
Pós-condições	

Tabela 12 – Regista Alunos

Caso de teste – Regista Alunos

Caso de Teste:	**Regista Alunos**
Objetivo:	Testar se os alunos são corretamente inseridos
Inputs	Dados do utilizador, dados dos alunos
Procedimentos:	1. Introduzir dados do administrador. 2. Introduzir os dados do aluno 3. Verificar se os dados inseridos correspondem aos dados esperados
Outputs	Dados dos alunos.

Tabela 13 – Caso de teste Regista Alunos

Caso de uso Cria turmas

Nome	Cria turmas
Atores	Administrador
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal efetuar a criação de turmas.
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Administrador escolhe a opção turmas. 2. Administrador seleciona a turma. 3. Administrador seleciona o aluno a inserir na turma. 4. Sistema guarda os dados da turma
Caminhos alternativos	3. a) Não há alunos. 4. a) O administrador cancela.
Suplementos ou Adornos	Testar se os alunos foram inseridos na turma corretamente
Pós-condições	

Tabela 14 – Cria Turmas

Caso de teste – Cria Turmas

Caso de Teste:	Cria Turmas
Objetivo:	Testar se os alunos são corretamente inseridos nas turmas
Inputs	Dados do utilizador, dados das turmas
Procedimentos:	1. Introduzir dados do administrador. 2. Introduzir aluno na turma 3. Verificar se os dados inseridos correspondem aos dados esperados
Outputs	Dados da turma.

Tabela 15 – Caso de teste – Cria Turmas

Caso de uso Configura tolerância

Nome	Configura tolerância
Atores	Administrador
Prioridade	Alta
Descrição	Este caso de uso tem como objetivo principal introduzir a tolerância permitida para os alunos introduzirem a assiduidade
Pré-condições	Tem que efetuar login
Caminho principal ou cenário principal	1. Administrador escolhe a opção Configurações. 2. Administrador introduz o valor da tolerância. 3. Sistema guarda o valor da tolerância.
Caminhos alternativos	2. a) Valor inválido. 5. a) O administrador cancela.
Suplementos ou Adornos	Testar o valor inserido está a ser utilizado na marcação de assiduidade.
Pós-condições	

Tabela 16 – Configura Tolerância

Caso de teste – Configura tolerância

Caso de Teste:	Configura tolerância
Objetivo:	Testar se a tolerância introduzida está a ser utilizada na marcação da assiduidade
Inputs	Dados do utilizador, dados da tolerância
Procedimentos:	1. Introduzir dados do administrador. 2. Introduzir valor da tolerância. 3. Verificar se o valor inserido está a ser conjugado com a data na inserção da assiduidade pelos alunos
Outputs	Dados de assiduidade

Tabela 17 – Caso de teste – Configura tolerância

4.3. Diagramas de Sequência

Os diagramas de sequência representam a interações entre objetos de um cenário, realizadas através de operações ou métodos. Dão enfase à ordem temporal em que os serviços são trocados entre objetos do sistema (Wikipédia, Diagrama de sequência, 2015). Em suma são utilizados para descrever o fluxo de execução dos casos de uso. Em cada diagrama estará representado os atores intervenientes bem como os objetos e métodos ordenados pelo tempo.

A figura 8 representa as ações do ator (Aluno) ao marcar a assiduidade no sistema.

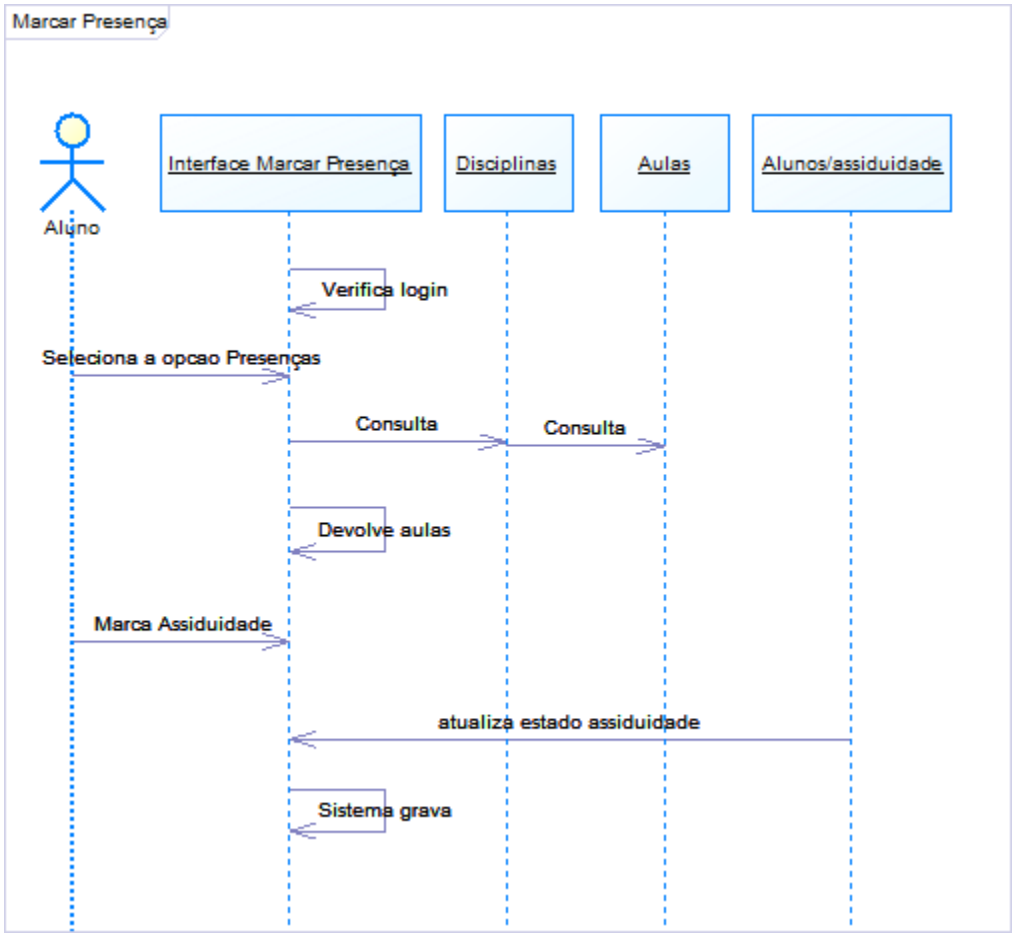

Figura 8 – Diagrama de sequência Marcar Presença

A Figura 9 representa as ações do ator aluno ao consultar a sua assiduidade na aplicação.

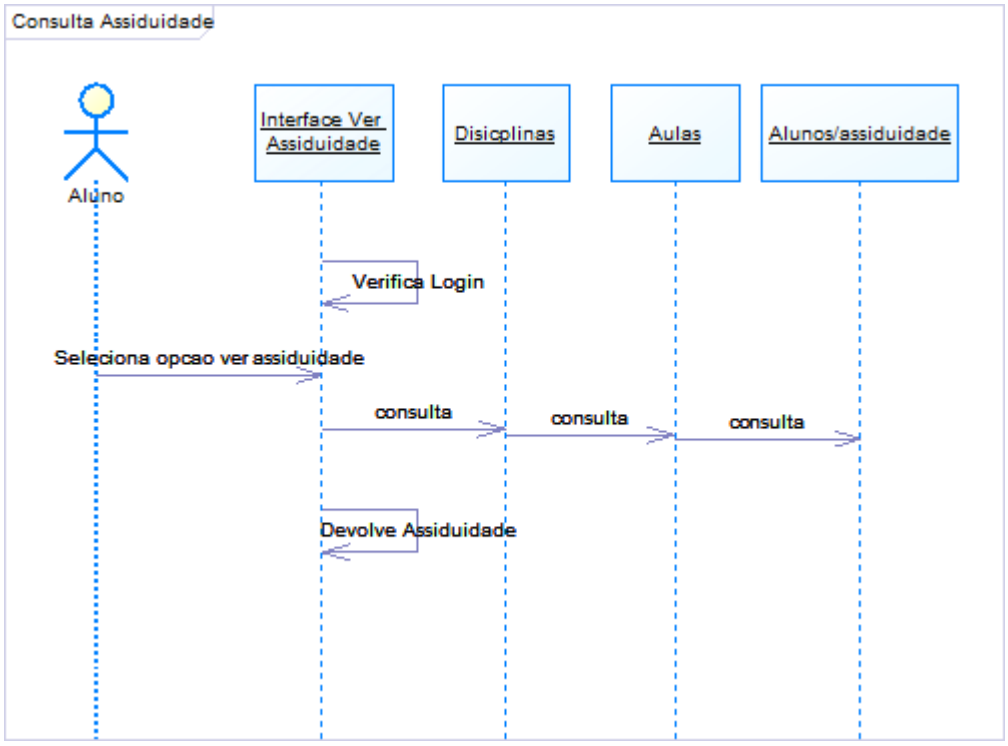

Figura 9 – Diagrama de sequência Consulta Assiduidade

Em baixo na Figura 10 estão representas as ações do professor ao verificar a assiduidade para a aprovar ou não.

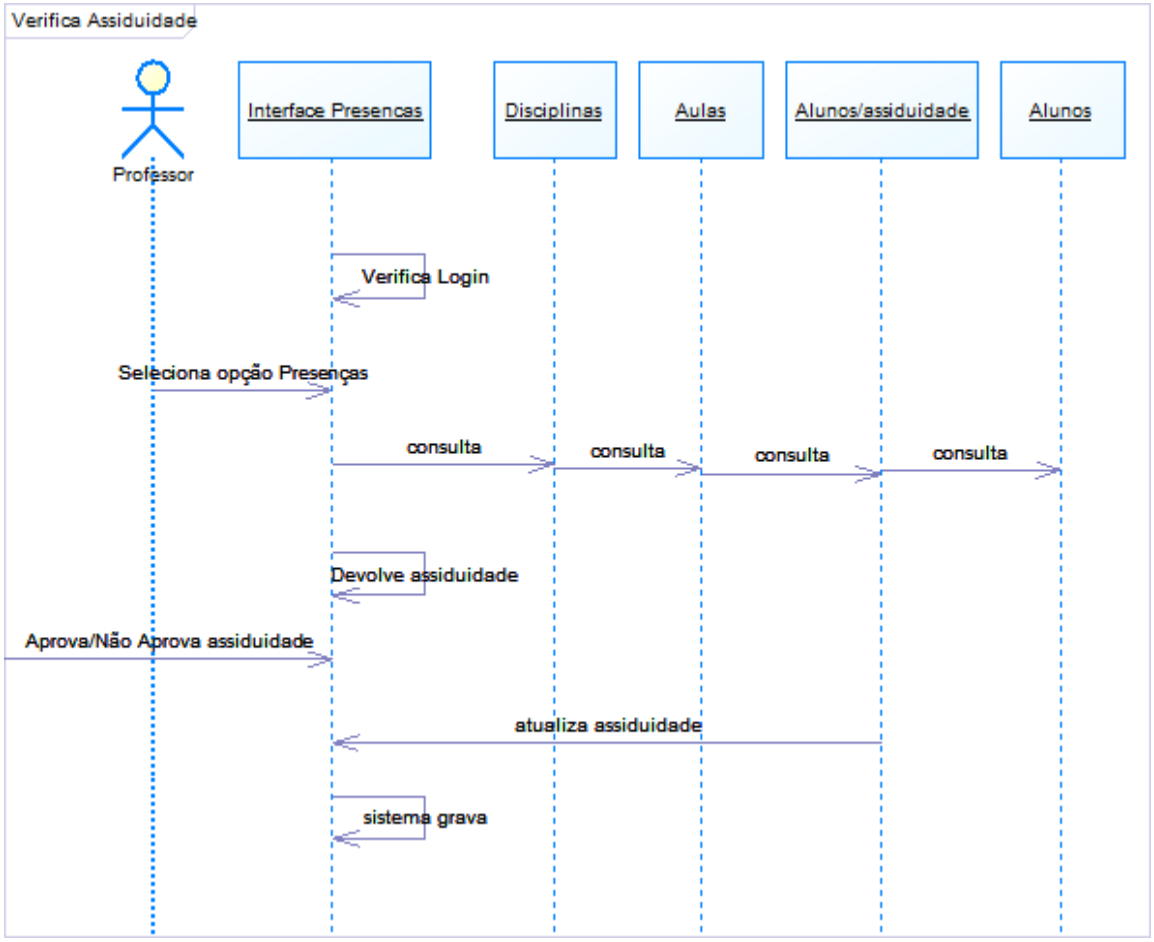

Figura 10 – Diagrama de sequência Verifica Assiduidade

No diagrama seguinte (Figura 11) estão representadas as ações necessárias que o professor necessita para marcar uma aula no sistema.

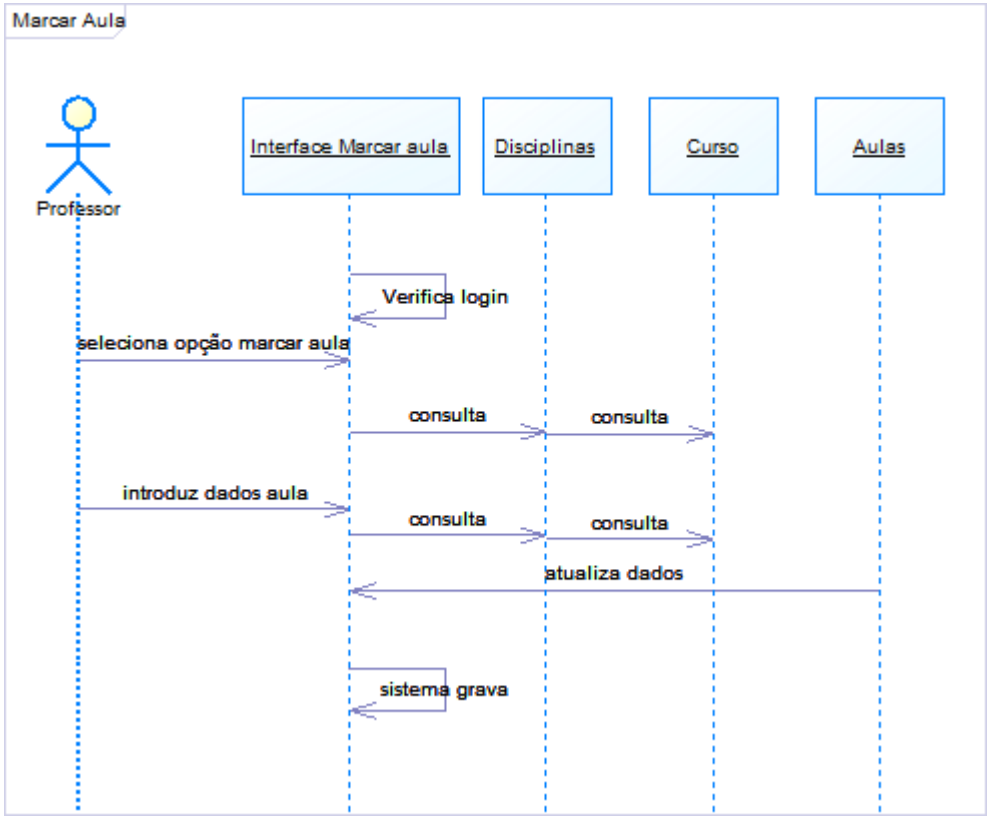

Figura 11 – Diagrama de sequência Marcar Aula

A Figura 12 representa as ações do ator Professor quando seleciona a opção relatórios na aplicação para poder consultar a assiduidade dos alunos

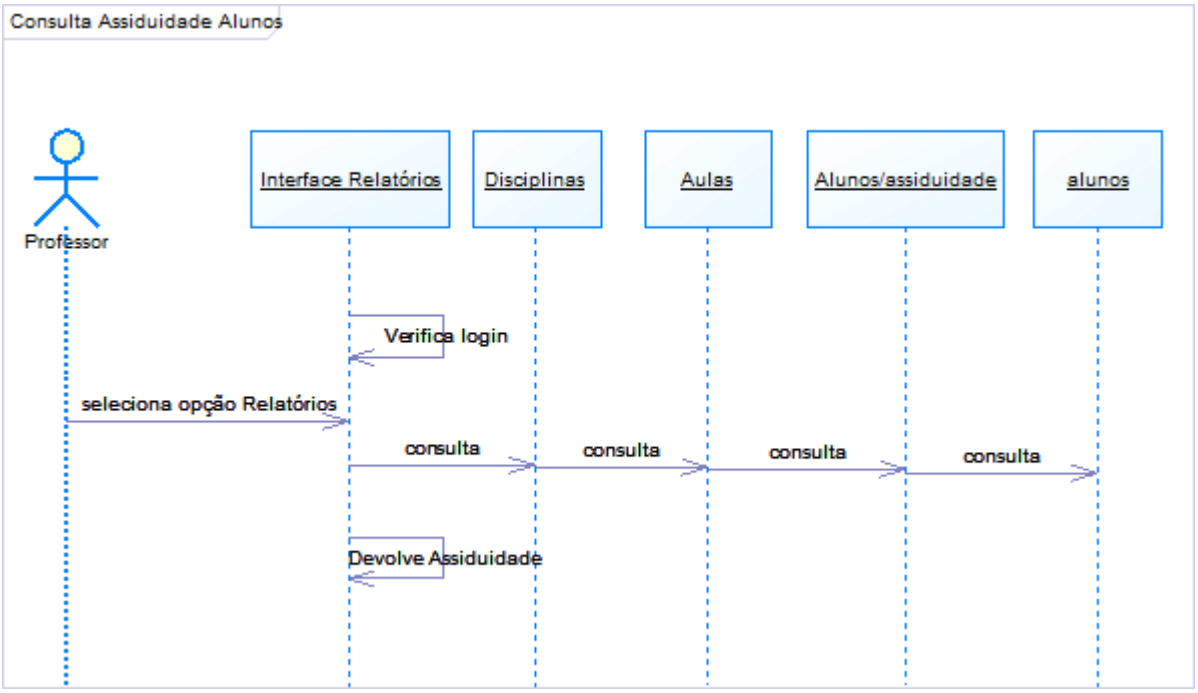

Figura 12 – Diagrama de sequência Consulta Assiduidade Alunos

O próximo diagrama (Figura 13) representa as ações do administrador quando introduz um novo aluno no sistema.

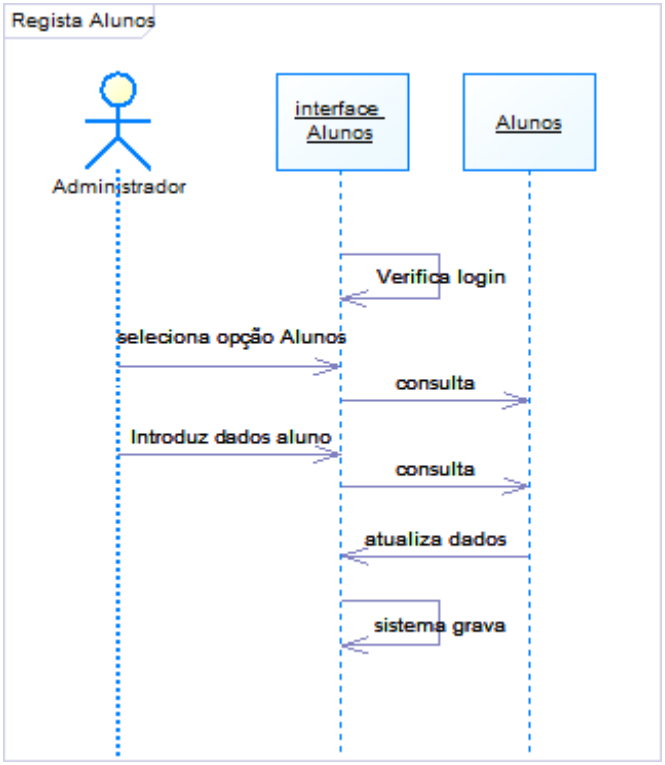

Figura 13 – Diagrama de sequência Regista alunos

O diagrama em baixo (Figura 14) representa as ações do administrador quando insere um aluno novo numa turma (disciplina) previamente criada.

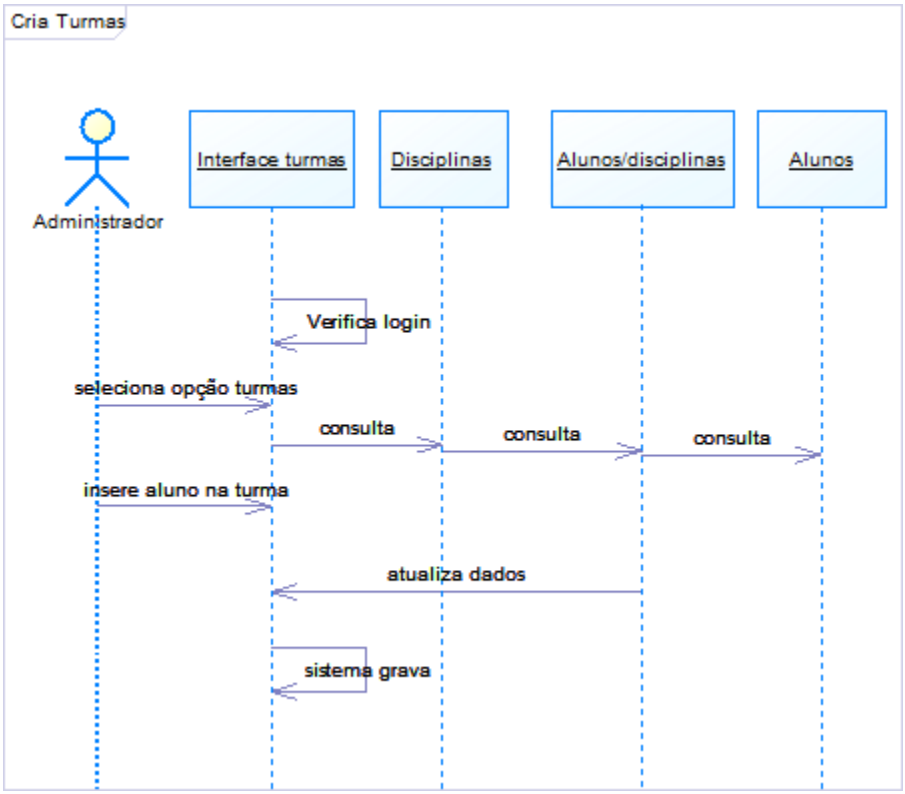

Figura 14 – Diagrama de sequência Cria Turmas

O próximo diagrama (Figura 15) representa as ações do ator administrador quando pretende configurar o tempo de tolerância para a marcação da assiduidade.

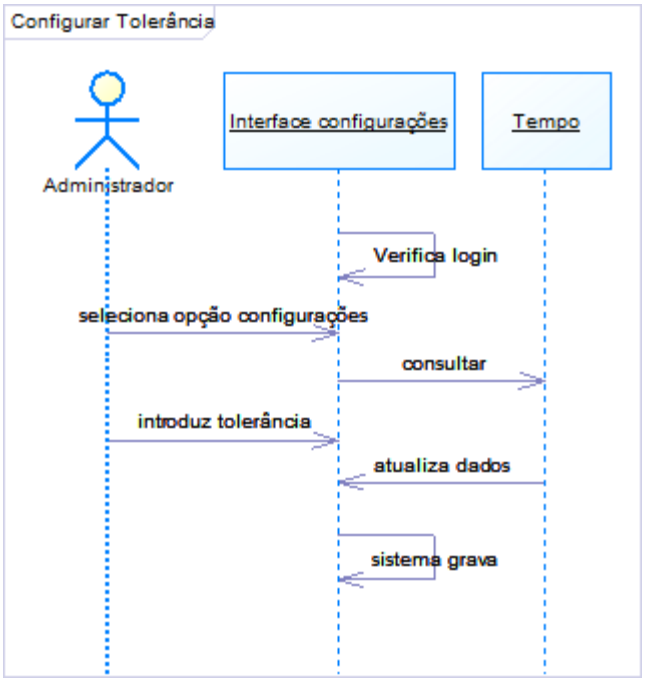

Figura 15 – Diagrama de sequência Configurar tolerância

4.4 Diagrama de Classes

O diagrama de classes é de elevadíssima importância uma vez que define a estrutura a desenvolver e mostra a relação entre as várias classes (Wikipédia, Diagrama de Classes, 2015). Na Figura 16 está representado o diagrama de classes deste projeto.

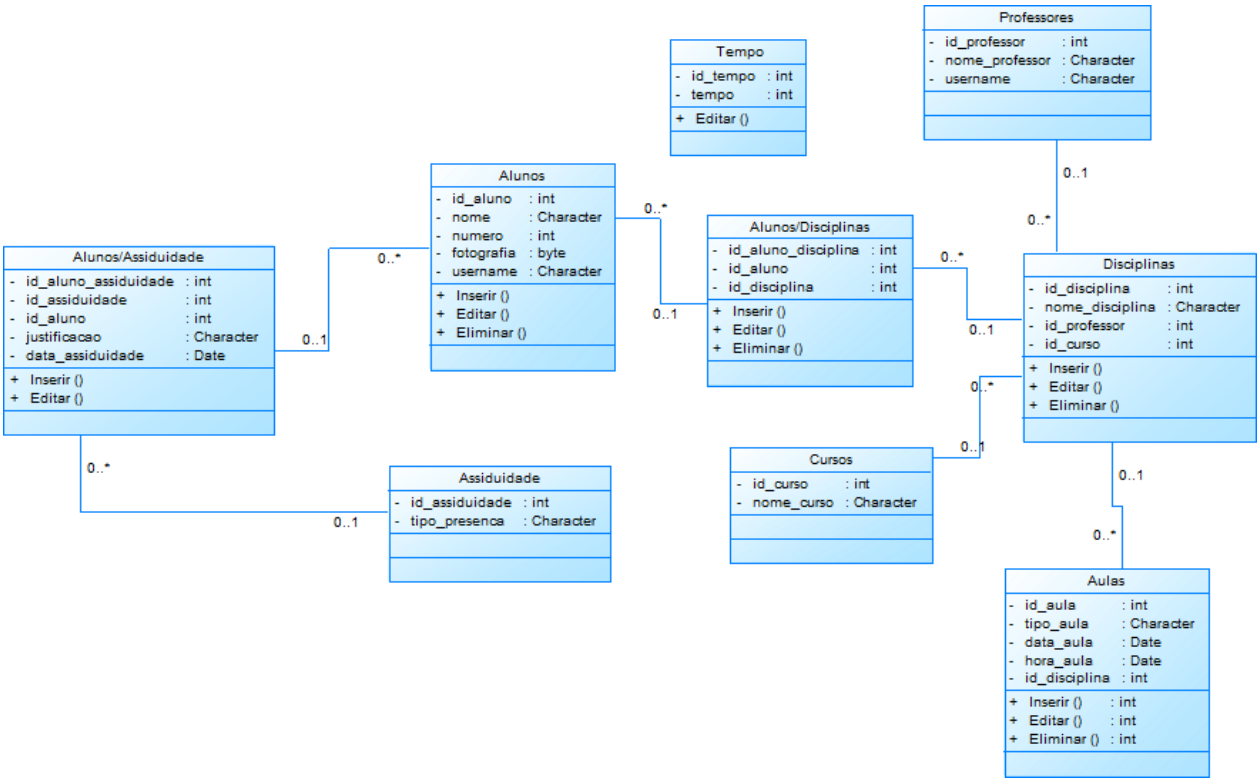

Figura 16 – Diagrama de Classes

4.5. Semântica de Classes

A semântica de classes é o conjunto de todos os atributos, tipos de dados, descrição, valores válidos, formato e restrições. Desta forma é possível saber rapidamente qualquer informação acerca de uma classe do projeto sendo por isso muito importante dispor dela.

4.5.1 Classe Alunos

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_aluno (PK)	Numeração automática	Número sequencial que identifica univocamente a cada aluno	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
nome	Nvarchar	String de caracteres que identifica o aluno	Caracteres de A a Z	Até 50 caracteres	Obrigatório
numero	Nvarchar	Número que identifica o número de aluno	Caracteres numericos	9 dígitos	Obrigatório
fotografia	image	Imagem que identifica o aluno	.jpeg		Não obrigatório
username	Nvarchar	String de caracteres que identifica o username do aluno	Caracteres de A a Z	Até 25 caracteres	Obrigatório

Tabela 18 – Classe Alunos

4.5.2 Classe Alunos/Assiduidade

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id (PK)	Numeração automática	Número sequencial que identifica univocamente o campo	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
id_assiduidade (FK)	int	Número sequencial que identifica univocamente a assiduidade	Maior que zero	Até 6 dígitos	Obrigatório
id_aluno (FK)	int	Número sequencial que identifica univocamente o aluno	Maior que zero	Até 6 dígitos	Obrigatório
justificacao	Nvarchar	String de caracteres que identifica a justificação de faltas	Caracteres de A a Z	Até 50 caracteres	Não obrigatório
data_assiduidade	date	Data da assiduidade	Dígitos de 0-9	dd-mm-aaaa	Obrigatório

Tabela 19 – Classe Alunos/Assiduidade

4.5.3 Classe Assiduidade

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_assiduidade (PK)	Numeração automática	Número sequencial que identifica univocamente o campo	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
tipo_presenca	Nvarchar	String de caracteres que identifica o tipo de presenças	Caracteres de A a Z	Até 50 caracteres	Obrigatório

Tabela 20 – Classe Assiduidade

4.5.4 Classe Alunos/Disciplinas

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_aluno_disciplina (PK)	Numeração automática	Número sequencial que identifica univocamente o campo	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
id_aluno (FK)	int	Número sequencial que identifica univocamente o aluno	Maior que zero	Até 6 dígitos	Obrigatório
id_disciplina (FK)	int	Número sequencial que identifica univocamente a disciplina	Maior que zero	Até 6 dígitos	Obrigatório

Tabela 21 – Classe Alunos/Disciplinas

4.5.5 Classe Disciplinas

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_disciplina (PK)	Numeração automática	Número sequencial que identifica univocamente o campo	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
nome_disciplina	Nvarchar	String de caracteres que identifica o nome da disciplina	Caracteres de A a Z	Até 50 caracteres	Obrigatório
id_professor	int	Número sequencial que identifica univocamente o professor	Maior que zero	Até 6 dígitos	Obrigatório
id_curso	int	Número sequencial que identifica univocamente o curso	Maior que zero	Até 6 dígitos	Obrigatório

Tabela 22 – Classe Disciplinas

4.5.6 Classe Aulas

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_aula (PK)	Numeração automática	Número sequencial que identifica univocamente o campo	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
tipo_aula	Nvarchar	String de caracteres que identifica o tipo de aula	Caracteres de A a Z	Até 50 caracteres	Obrigatório
data_aula	date	Data da aula	Dígitos de 0-9	dd-mm-aaaa	Obrigatório
hora_aula	date	Hora da aula	Dígitos de 0-9	00:00:00	Obrigatório

Tabela 23 – Classe Aulas

4.5.7 Classe Professores

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_professor(PK)	Numeração automática	Número sequencial que identifica univocamente o professor	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
nome_professor	Nvarchar	String de caracteres que identifica o nome do professor	Caracteres de A a Z	Até 50 caracteres	Obrigatório
username	Nvarchar	String de caracteres que identifica o username do professor	Caracteres de A a Z	Até 50 caracteres	Obrigatório

Tabela 24 – Classe Professores

4.5.8 Classe Cursos

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_curso(PK)	Numeração automática	Número sequencial que identifica univocamente o professor	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
nome_curso	Nvarchar	String de caracteres que identifica o nome do curso	Caracteres de A a Z	Até 50 caracteres	Obrigatório

Tabela 25 – Classe Cursos

4.5.9 Classe Tempo

Nome	Tipo de dados	Descrição	Valores válidos	Formato	Restrições
id_tempo (PK)	Numeração automática	Número sequencial que identifica univocamente o professor	Maior que zero	Até 6 dígitos	Criado pelo sistema e não alterável
tempo	int	Valor da tolerância para a assiduidade	Maior que zero	Até 2 dígitos	Obrigatório

Tabela 26 – Classe Tempo

5. Implementação da Solução

Depois do estudo e levantamento dos requisitos necessários à elaboração do projeto, foi mais fácil passar à sua implementação. Para o desenvolvimento deste projeto foi usada a ferramenta da Microsoft Visual Studio 2012 no desenvolvimento da aplicação em si. Quanto ao layout da aplicação foi utilizada a ferramenta Bootstrap, como referido acima no capítulo das Metodologias. O Bootstrap facilitou imenso na construção da aplicação, pois tem muitas configurações e componentes pré-configurados que facilitam a construção de um layout, neste caso um layout responsivo. Para a construção da base de dados do projeto foi utilizada o SGBD da Microsoft, SQL Server 2012. De seguida irão ser abordadas algumas interfaces que considero mais importantes no projeto acompanhadas por excertos de código, e também o diagrama ER utilizado na construção da base de dados do projeto.

5.1 Base de dados

A Figura 17 representa o modelo físico deste projeto, representa a base de dados utilizada no desenvolvimento da aplicação web.

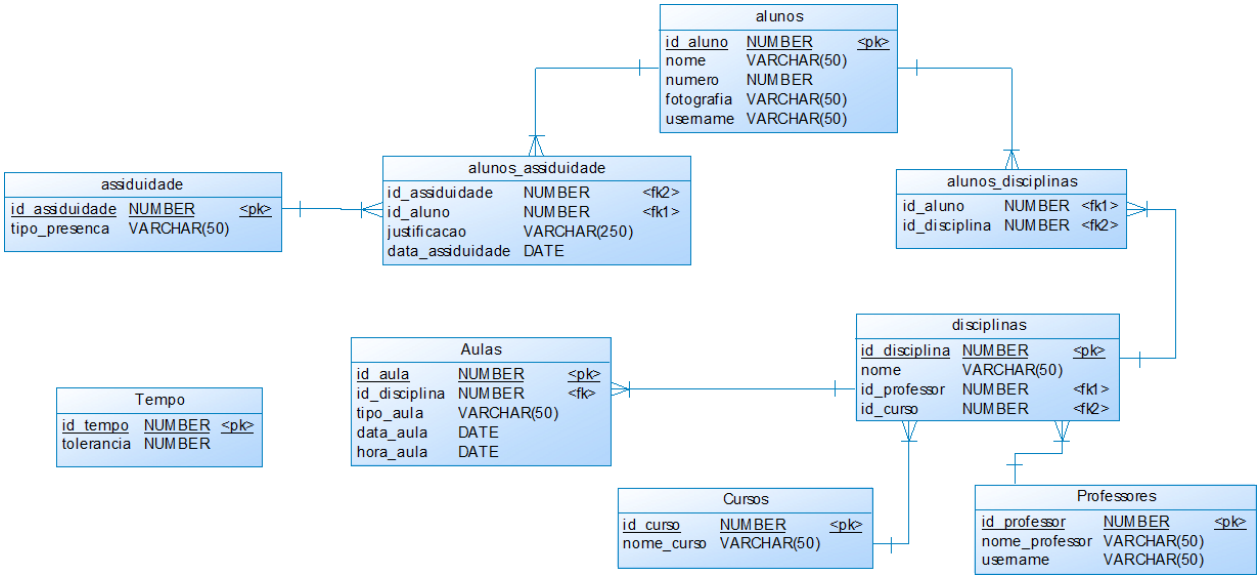

Figura 17 – Modelo ER

O modelo físico foi elaborado na fase final do desenvolvimento pois foi diversas vezes alterado ao longo do projeto quando necessário. A Figura 17 representa a versão final do modelo físico.

5.2 Interface Marcar Presença

A interface representada na Figura 18 é das mais importantes no projeto. É a interface que permite ao aluno marcar a sua assiduidade na aula. Se a aula já passou, ou se está fora dos limites de tolerância definidos, não é possível ao aluno a marcação da assiduidade.

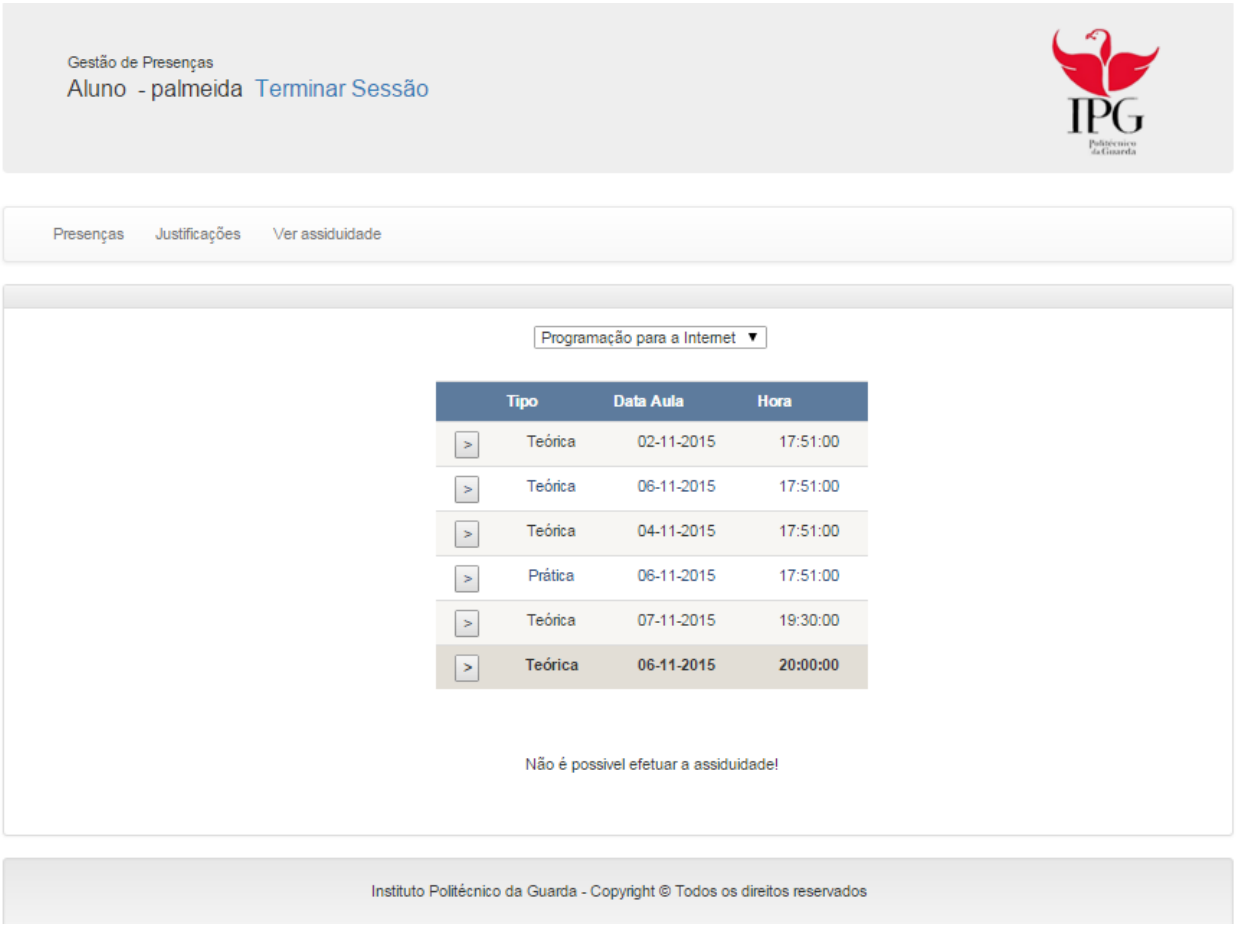

Figura 18 – Interface Marcar Presença

5.2.1 Código – Interface Marcar Presença

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Globalization;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class marcarpresenca : System.Web.UI.Page
{
    protected string conS = "Data Source=PEDRO-
PORTATIL\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\BD_GP.mdf;Integrated
Security=True";
    protected SqlConnection con;
    public string dataaula;
    public string tolerancia;
    public Boolean inserir = false;
    protected void Page_Load(object sender, EventArgs e)
    {
        Panel1.Visible = false;
        Panel2.Visible = false;
        SqlDataSourceaulas.SelectParameters["user"].DefaultValue =
User.Identity.Name;
        SqlDataSource1.SelectParameters["user"].DefaultValue =
User.Identity.Name;

        con = new SqlConnection(conS);
        con.Open();
        string q = "SELECT * from tempo";
        SqlCommand query = new SqlCommand(q,con);
        SqlDataReader dr = query.ExecuteReader();
        while (dr.Read())
        {

            Label1.Text = dr["tempo_tolerancia"].ToString();//vai buscar tempo
tolerancia definido
        }
        tolerancia = Label1.Text;

    }
    protected void Buttonmarcarpresenca_Click(object sender, EventArgs e)
    {
        if (inserir == false)
        {
            PanelDisciplinasPresencas.Visible = false;
            FormViewAssiduidade.ChangeMode(FormViewMode.Insert);
        }
        else {
            Panel2.Visible = true;
        }
```

```
    }

    protected void FormViewAssiduidade_ItemCreated(object sender, EventArgs e)
    {
        GridView1.DataBind();
    }
    protected void FormViewAssiduidade_ModeChanged(object sender, EventArgs e)
    {
        if (FormViewAssiduidade.CurrentMode == FormViewMode.ReadOnly)
        {
            PanelDisciplinasPresencas.Visible = true;
        }
        else
        {
            PanelDisciplinasPresencas.Visible = false;
        }
    }

    protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
    {
        dataaula = GridView1.SelectedRow.Cells[3].Text; //vai buscar hora da
aula
        DateTime da = Convert.ToDateTime(dataaula);

        int tt = Convert.ToInt32(tolerancia.ToString());

        TimeSpan span = DateTime.Now-da;

        int horas = span.Hours;
        int minutos = span.Minutes;
        DateTime dataatual = DateTime.Now;
        int minutosatuais = dataatual.Minute;
        string diaaula = GridView1.SelectedRow.Cells[2].Text;
        DateTime daula = Convert.ToDateTime(diaaula);
        int diadaaula = daula.Day;
        int diaatual = dataatual.Day;

        if ((diaatual == diadaaula) && (horas == 0) && (minutos <= tt) &&
(minutosatuais >= da.Minute))
        {
            if (GridView1.SelectedIndex >= 0)
            {
                Buttonmarcarpresenca.Visible = true;

            }
            else
            {
                Buttonmarcarpresenca.Visible = false;
            }
        }
        else {
            Panel1.Visible = true;
            Buttonmarcarpresenca.Visible = false;
```

```
        }
    }
    private TextBox id_assiduidadeTextBox = null;
    private DropDownList DropDownList2 = null;
    protected void id_assiduidadeTextBox_Load(object sender, EventArgs e)
    {
        id_assiduidadeTextBox = (TextBox)sender;
    }
    protected void DropDownList2_Load(object sender, EventArgs e)
    {
        DropDownList2 = (DropDownList)sender;
    }
    protected void DropDownList2_TextChanged(object sender, EventArgs e)
    {
        if (DropDownList2.Text != id_assiduidadeTextBox.Text)
        {
            id_assiduidadeTextBox.Text = DropDownList2.Text;
        }
    }
    protected void id_assiduidadeTextBox_DataBinding(object sender, EventArgs
e)
    {
        if (DropDownList2.Text != id_assiduidadeTextBox.Text)
        {
            DropDownList2.Text = id_assiduidadeTextBox.Text;
        }
    }

    private TextBox id_alunoTextBox = null;
    private DropDownList DropDownList3 = null;
    protected void id_alunoTextBox_Load(object sender, EventArgs e)
    {
        id_alunoTextBox = (TextBox)sender;
    }
    protected void DropDownList3_Load(object sender, EventArgs e)
    {
        DropDownList3 = (DropDownList)sender;
    }
    protected void DropDownList3_TextChanged(object sender, EventArgs e)
    {
        if (DropDownList3.Text != id_alunoTextBox.Text)
        {
            id_alunoTextBox.Text = DropDownList3.Text;
        }
    }
    protected void id_alunoTextBox_DataBinding(object sender, EventArgs e)
    {
        if (DropDownList3.Text != id_alunoTextBox.Text)
        {
            DropDownList3.Text = id_alunoTextBox.Text;
        }
    }
```

```
    protected void ButtonInserirAssiduidade_Click(object sender, EventArgs e)
    {
        inserir = true;
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Panel2.Visible = false;
    }
}
```

5.3 Interface Validar Presença

A interface representada na Figura 19 é também uma das mais importantes neste projeto, pois é nela que o professor valida ou não a assiduidade.

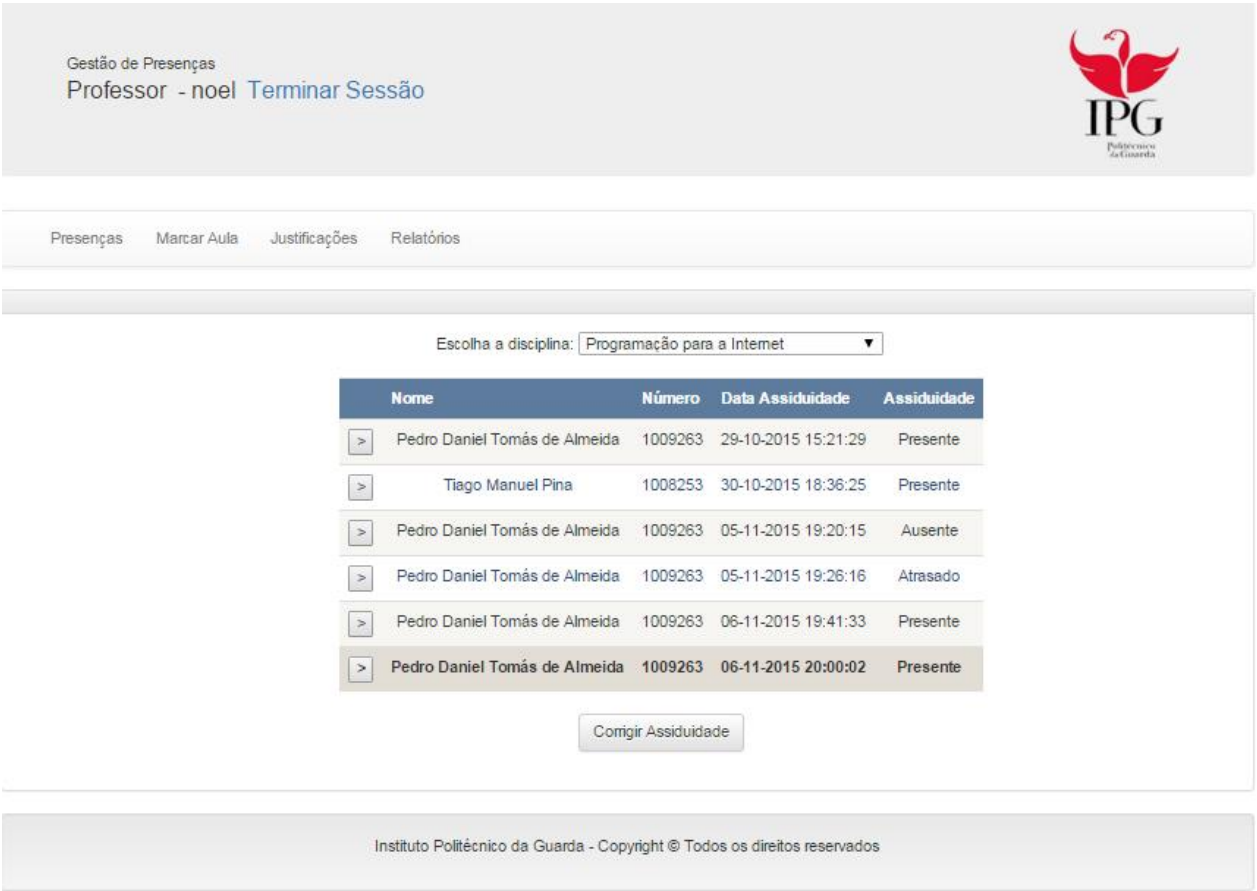

Figura 19 – Interface Validar Assiduidade

5.3.1 Código – Interface Validar Presença

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class presencas : System.Web.UI.Page
{

    protected void Page_Load(object sender, EventArgs e)
    {
        SqlDataSourceDisciplina.SelectParameters["user"].DefaultValue =
User.Identity.Name;
        ButtonCorrigirAssiduidade.Visible = false;
        if (GridViewAlunos.Rows.Count > 0)
        {

            ButtonCorrigirAssiduidade.Visible = true;

        }
        else
        {
            ButtonCorrigirAssiduidade.Visible = false;
        }

    }
    protected void GridViewAlunos_SelectedIndexChanged(object sender,
EventArgs e)
    {
        if (GridViewAlunos.SelectedIndex >= 0)
        {
            ButtonCorrigirAssiduidade.Visible = true;


        }
        else
        {
            ButtonCorrigirAssiduidade.Visible = false;

        }
    }

    protected void ButtonCorrigirAssiduidade_Click(object sender, EventArgs e)
    {
        ButtonCorrigirAssiduidade.Visible = false;
        FormViewPresencas.ChangeMode(FormViewMode.Edit);

    }
```

```
    protected void FormViewPresencas_ModeChanged(object sender, EventArgs e)
    {
        if (FormViewPresencas.CurrentMode == FormViewMode.ReadOnly)
        {
            PanelPresenças.Visible = true;
        }
        else
        {
            PanelPresenças.Visible = false;
        }
    }

    protected void FormViewPresencas_ItemUpdated(object sender,
FormViewUpdatedEventArgs e)
    {
        GridViewAlunos.DataBind();
    }
    private TextBox id_assiduidadeTextBox = null;
    private DropDownList DropDownList1 = null;
    protected void id_assiduidadeTextBox_Load(object sender, EventArgs e)
    {
        id_assiduidadeTextBox = (TextBox)sender;
    }
    protected void DropDownList1_Load(object sender, EventArgs e)
    {
        DropDownList1 = (DropDownList)sender;
    }
    protected void DropDownList1_TextChanged(object sender, EventArgs e)
    {
        if (DropDownList1.Text != id_assiduidadeTextBox.Text)
        {
            id_assiduidadeTextBox.Text = DropDownList1.Text;
        }
    }
    protected void id_assiduidadeTextBox_DataBinding(object sender, EventArgs
e)
    {
        if (DropDownList1.Text != id_assiduidadeTextBox.Text)
        {
            DropDownList1.Text = id_assiduidadeTextBox.Text;
        }
    }
}
```

5.4 Interface Turmas

A interface representada na Figura 20 permite ao administrador a criação de turmas, é também uma das interfaces mais importantes deste projeto.

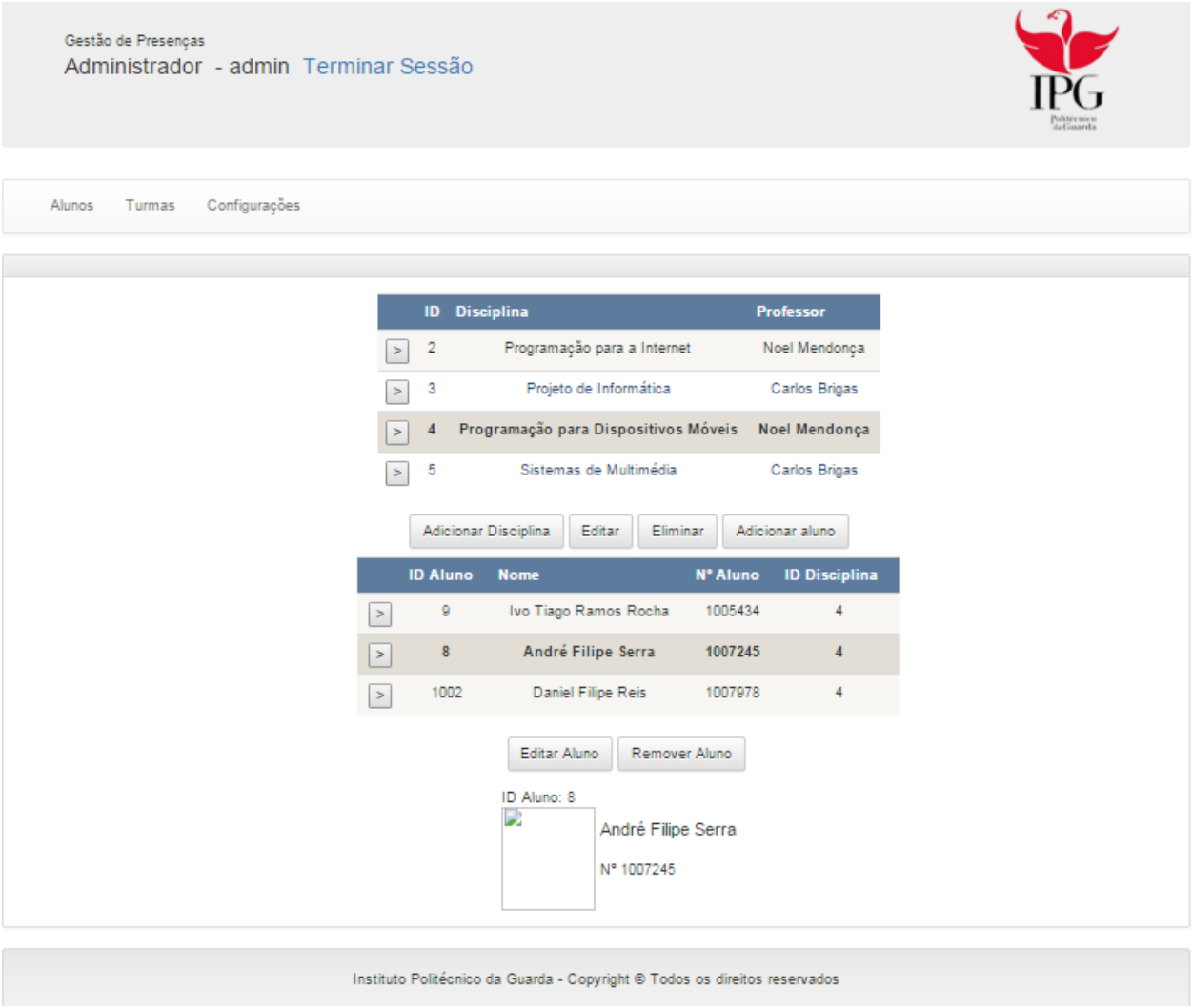

Figura 20 – Interface Turmas

5.4.1 Código – Interface Turmas

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class turmas : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (GridViewAlunos.Rows.Count > 0)
        {

            btEditAlunoDisciplina.Visible = true;
            btRemoveAlunoDisciplina.Visible = true;

        }
        else
        {

            btEditAlunoDisciplina.Visible = false;
            btRemoveAlunoDisciplina.Visible = false;

        }
    }
    protected void btAdicionarDisciplina_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = false;
        PanelAlunos.Visible = false;
        FormViewDisciplinas.ChangeMode(FormViewMode.Insert);
    }
    protected void ButtonAdicionarAlunoDisciplina_Click(object sender,
EventArgs e)
    {
        PanelDisciplinas.Visible = false;
        FormViewAdicionarAluno.ChangeMode(FormViewMode.Insert);
    }
    protected void BtCancel_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = true;
    }
    protected void BtCancelar_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = true;
    }
    protected void btEditardisciplina_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = false;
        FormViewDisciplinas.ChangeMode(FormViewMode.Edit);
    }
```

```
    protected void GridViewDisciplinas_SelectedIndexChanged(object sender,
EventArgs e)
    {
        if (GridViewDisciplinas.SelectedIndex >= 0)
        {
            btEditardisciplina.Visible = true;
            btEliminardisciplina.Visible = true;
            ButtonAdicionarAlunoDisciplina.Visible = true;

        }
        else
        {
            btEditardisciplina.Visible = false;
            btEliminardisciplina.Visible = false;
            ButtonAdicionarAlunoDisciplina.Visible = false;
        }
    }

    protected void FormViewDisciplinas_ItemCreated(object sender, EventArgs e)
    {
        GridViewDisciplinas.DataBind();
    }
    protected void FormViewDisciplinas_ItemDeleted(object sender,
FormViewDeletedEventArgs e)
    {
        GridViewDisciplinas.DataBind();
    }
    protected void FormViewDisciplinas_ItemUpdated(object sender,
FormViewUpdatedEventArgs e)
    {
        GridViewDisciplinas.DataBind();
    }

    protected void FormViewDisciplinas_ModeChanged(object sender, EventArgs e)
    {
        if (FormViewDisciplinas.CurrentMode == FormViewMode.ReadOnly)
        {
            PanelDisciplinas.Visible = true;
            PanelAlunos.Visible = true;
        }
        else
        {
            PanelDisciplinas.Visible = false;
            PanelAlunos.Visible = false;
        }
    }

    protected void FormViewAdicionarAluno_ItemCreated(object sender, EventArgs
e)
    {
        GridViewDisciplinas.DataBind();
        GridViewAlunos.DataBind();
    }
```

```
    protected void FormViewAdicionarAluno_ItemDeleted(object sender,
FormViewDeletedEventArgs e)
    {
        GridViewDisciplinas.DataBind();
        GridViewAlunos.DataBind();
    }
    protected void FormViewAdicionarAluno_ItemUpdated(object sender,
FormViewUpdatedEventArgs e)
    {
        GridViewDisciplinas.DataBind();
    }

    protected void FormViewAdicionarAluno_ModeChanged(object sender, EventArgs
e)
    {
        if (FormViewAdicionarAluno.CurrentMode == FormViewMode.ReadOnly)
        {
            PanelDisciplinas.Visible = true;
        }
        else
        {
            PanelDisciplinas.Visible = false;
        }
    }

    protected void btEliminardisciplina_Click(object sender, EventArgs e)
    {
        if (GridViewAlunos.Rows.Count > 0)
        {
            PanelDisciplinas.Visible = false;
            PanelAviso.Visible = true;
            PanelAlunos.Visible = false;
        }
        else
        {
            PanelDisciplinas.Visible = false;
            PanelAlunos.Visible = false;
            PanelApagar.Visible = true;
        }




    }
    protected void btCancelar_Click(object sender, EventArgs e)
    {
        PanelAlunos.Visible = true;
        PanelDisciplinas.Visible = true;
        PanelApagar.Visible = false;
    }
    protected void btApagar_Click(object sender, EventArgs e)
    {
        FormViewDisciplinas.DeleteItem();
```

```
        PanelApagar.Visible = false;
        PanelDisciplinas.Visible = true;
    }


    private TextBox tbAluno = null;
    private TextBox tbDisciplina = null;
    protected void tbAluno_Load(object sender, EventArgs e)
    {

        tbAluno = (TextBox)sender;
    }

    protected void tbDisciplina_Load(object sender, EventArgs e)
    {

        tbDisciplina = (TextBox)sender;
    }

    private DropDownList DropDownListAlunos = null;
    private DropDownList DropDownListDisciplina = null;
    protected void DropDownListAlunos_Load(object sender, EventArgs e)
    {
        DropDownListAlunos = (DropDownList)sender;
    }

    protected void DropDownListDisciplina_Load(object sender, EventArgs e)
    {
        DropDownListDisciplina = (DropDownList)sender;
    }

    protected void DropDownListAlunos_TextChanged(object sender, EventArgs e)
    {
        if (DropDownListAlunos.Text != tbAluno.Text)
        {
            tbAluno.Text = DropDownListAlunos.Text;
        }
    }

    protected void DropDownListDisciplina_TextChanged(object sender, EventArgs
e)
    {
        if (DropDownListDisciplina.Text != tbDisciplina.Text)
        {
            tbDisciplina.Text = DropDownListDisciplina.Text;
        }
    }

    protected void tbAluno_DataBinding(object sender, EventArgs e)
    {
        if (DropDownListAlunos.Text != tbAluno.Text)
        {
            DropDownListAlunos.Text = tbAluno.Text;
```

```
        }
    }

    protected void tbDisciplina_DataBinding(object sender, EventArgs e)
    {
        if (DropDownListDisciplina.Text != tbDisciplina.Text)
        {
            DropDownListDisciplina.Text = tbDisciplina.Text;
        }
    }

    private DropDownList DropDownListProfessor = null;
    private TextBox TextBoxProfessor = null;
    protected void TextBoxProfessor_Load(object sender, EventArgs e)
    {
        TextBoxProfessor = (TextBox)sender;
    }
    protected void DropDownListProfessor_Load(object sender, EventArgs e)
    {
        DropDownListProfessor = (DropDownList)sender;
    }

    protected void DropDownListProfessor_TextChanged(object sender, EventArgs
e)
    {
        if (DropDownListProfessor.Text != TextBoxProfessor.Text)
        {
            TextBoxProfessor.Text = DropDownListProfessor.Text;
        }
    }

    protected void TextBoxProfessor_DataBinding(object sender, EventArgs e)
    {
        if (DropDownListProfessor.Text != TextBoxProfessor.Text)
        {
            DropDownListProfessor.Text = TextBoxProfessor.Text;
        }
    }
    protected void btEditAlunoDisciplina_Click(object sender, EventArgs e)
    {
        PanelAlunos.Visible = false;
        PanelDisciplinas.Visible = false;
        FormViewAlunos.ChangeMode(FormViewMode.Edit);
    }

    protected void FormViewAlunos_ItemUpdated(object sender,
FormViewUpdatedEventArgs e)
    {
        GridViewAlunos.DataBind();
    }

    protected void FormViewAlunos_ItemDeleted(object sender,
FormViewDeletedEventArgs e)
```

```
    {
        GridViewAlunos.DataBind();
    }

    private FileUpload fileUploadFoto = null;
    protected void FileUploadImagem_Load(object sender, EventArgs e)
    {
        fileUploadFoto = (FileUpload)sender;
    }

    protected void FormViewAlunos_ItemUpdating(object sender,
FormViewUpdateEventArgs e)
    {
        {
            if (fileUploadFoto.HasFile)
            {
                e.NewValues["fotografia"] = fileUploadFoto.FileBytes;
            }
            else
            {
                SqlDataSourceFormAlunos.UpdateCommand = "UPDATE alunos SET
nome = @nome, numero = @numero, username = @username WHERE (id_aluno =
@id_aluno)";
            }
        }
    }

    protected void FormViewAlunos_ModeChanged(object sender, EventArgs e)
    {
        if (FormViewDisciplinas.CurrentMode == FormViewMode.ReadOnly)
        {
            PanelDisciplinas.Visible = true;
            PanelAlunos.Visible = true;
        }
        else
        {
            PanelDisciplinas.Visible = false;
            PanelAlunos.Visible = false;
        }
    }

    protected void btCancelarAluno_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = true;
        PanelAlunos.Visible = true;
        PanelApagarAluno.Visible = false;


    }
    protected void btApagarAluno_Click(object sender, EventArgs e)
    {
        FormViewAlunos.DeleteItem();
        //FormViewAdicionarAluno.DeleteItem();
```

```
        PanelApagarAluno.Visible = false;
        PanelDisciplinas.Visible = true;
        PanelAlunos.Visible = true;
    }


    protected void btRemoveAlunoDisciplina_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = false;
        PanelAlunos.Visible = false;
        PanelApagarAluno.Visible = true;
    }
    protected void Button3_Click(object sender, EventArgs e)
    {
        PanelDisciplinas.Visible = true;
        PanelApagar.Visible = false;
        PanelAlunos.Visible = true;
        PanelAviso.Visible = false;
    }

}
```

5.5 Interface Marcar Aula

A interface representada na Figura 21 permite ao professor a marcação de uma aula, para posteriormente os alunos marcarem a assiduidade nessa mesma aula.

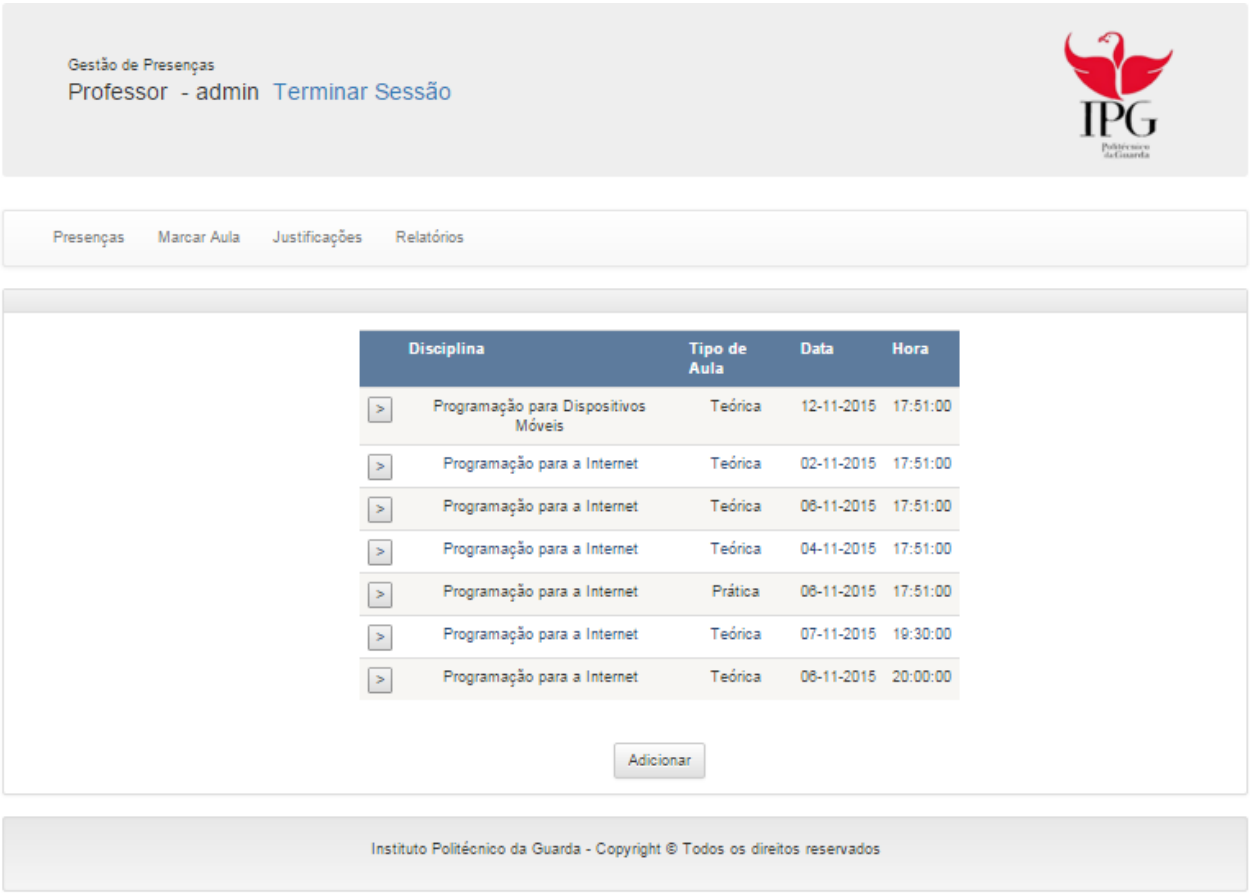

Figura 21 – Interface Marcar Aula

5.5.1 Código – Interface Marcar Aula

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.Security;

public partial class marcaraula : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

        SqlDataSource1.SelectParameters["user"].DefaultValue =
User.Identity.Name;
    }
    protected void Buttonadd_Click(object sender, EventArgs e)
    {
        PanelAulas.Visible = false;
        FormViewAulas.ChangeMode(FormViewMode.Insert);
    }
    protected void Buttonedit_Click(object sender, EventArgs e)
    {
        PanelAulas.Visible = false;
        FormViewAulas.ChangeMode(FormViewMode.Edit);
    }
    protected void Buttonremove_Click(object sender, EventArgs e)
    {
        PanelAulas.Visible = false;
        PanelApagar.Visible = true;
    }
    protected void btCancelar_Click(object sender, EventArgs e)
    {
        PanelAulas.Visible = true;
        PanelApagar.Visible = false;
    }
    protected void btApagar_Click(object sender, EventArgs e)
    {
        FormViewAulas.DeleteItem();
        PanelApagar.Visible = false;
        PanelAulas.Visible = true;
    }

    protected void FormViewAulas_ItemCreated(object sender, EventArgs e)
    {
        GridViewAulas.DataBind();
    }
    protected void FormViewAulas_ItemDeleted(object sender,
FormViewDeletedEventArgs e)
    {
        GridViewAulas.DataBind();
```

```
    }
    protected void FormViewAulas_ItemUpdated(object sender,
FormViewUpdatedEventArgs e)
    {
        GridViewAulas.DataBind();
    }

    protected void FormViewAulas_ModeChanged(object sender, EventArgs e)
    {
        if (FormViewAulas.CurrentMode == FormViewMode.ReadOnly)
        {
            PanelAulas.Visible = true;
        }
        else
        {
            PanelAulas.Visible = false;
        }
    }

    protected void GridViewAulas_SelectedIndexChanged(object sender, EventArgs
e)
    {
        if (GridViewAulas.SelectedIndex >= 0)
        {
            Buttonedit.Visible = true;
            Buttonremove.Visible = true;

        }
        else
        {
            Buttonedit.Visible = false;
            Buttonremove.Visible = false;
        }
    }

    private DropDownList DropDownListDisciplina = null;
    private TextBox TextBoxDisciplina = null;
    protected void TextBoxDisciplina_Load(object sender, EventArgs e)
    {
        TextBoxDisciplina = (TextBox)sender;
    }
    protected void DropDownListDisciplina_Load(object sender, EventArgs e)
    {
        DropDownListDisciplina = (DropDownList)sender;
    }

    protected void DropDownListDisciplina_TextChanged(object sender, EventArgs
e)
    {
        if (DropDownListDisciplina.Text != TextBoxDisciplina.Text)
        {
            TextBoxDisciplina.Text = DropDownListDisciplina.Text;
        }
```

```
    }

    protected void TextBoxDisciplina_DataBinding(object sender, EventArgs e)
    {
        if (DropDownListDisciplina.Text != TextBoxDisciplina.Text)
        {
            DropDownListDisciplina.Text = TextBoxDisciplina.Text;
        }
    }
}
```

6. Conclusões

Ao longo da elaboração deste projeto foi possível trabalhar com tecnologias que já tinha explorado na licenciatura em Engenharia Informática, nomeadamente a plataforma da Microsoft ASP.NET, na linguagem HTML, C# e CSS, embora não tivesse um conhecimento aprofundado, este projeto permitiu-me enriquecer mais esses conhecimentos.

O desenvolvimento para a web está sempre em crescimento e em mudança, o que me levou a ter que estudar e procurar alternativas ao que tinha aprendido no passado ao longo da licenciatura. Um exemplo disso foi como fazer um layout de uma página web responsiva. Foi algo que não aprendi durante a licenciatura pois os dispositivos móveis ainda não eram populares como são hoje em dia. Este problema foi ultrapassado através da ajuda do Bootstrap, um conjunto de ferramentas muito útil no desenvolvimento de aplicações web.

Durante a implementação da aplicação foram surgindo outras questões. Um exemplo disso foi o desenvolvimento de um dos objetivos, como fazer para que os alunos só conseguissem marcar a assiduidade na hora da aula. Esta questão foi rapidamente ultrapassada com a ajuda do meu orientador. Em relação aos objetivos inicialmente previstos, pode-se dizer que foram cumpridos.

No geral penso que o desenvolvimento deste projeto me enriqueceu numa área que pessoalmente gosto de desenvolver.

7. Bibliografia

Grando, N. (11 de Junho de 2015). *Metodologias Ágeis no Desenvolvimento de Projetos de software*. Obtido de Blog do Nei: https://neigrando.wordpress.com/2010/09/06/metodologias-ageis-no-desenvolvimento-de-projetos-de-software/

Wikipedia. (25 de 11 de 2015). *Bootstrap (front-end framework)*. Obtido de Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)

Wikipédia. (10 de 10 de 2015). *Diagrama de caso de uso*. Obtido de Wikipédia, a enciclopédia livre: https://pt.wikipedia.org/wiki/Diagrama_de_caso_de_uso

Wikipédia. (15 de 10 de 2015). *Diagrama de Classes*. Obtido de Wikipédia, a enciclopédia livre: https://pt.wikipedia.org/wiki/Diagrama_de_classes

Wikipédia. (4 de 10 de 2015). *Diagrama de contexto*. Obtido de Wikipédia, a enciclopédia livre: https://pt.wikipedia.org/wiki/Diagrama_de_contexto

Wikipédia. (15 de 10 de 2015). *Diagrama de sequência*. Obtido de Wikipédia, a enciclopédia livre: https://pt.wikipedia.org/wiki/Diagrama_de_sequ%C3%AAncia

Wikipédia. (15 de 10 de 2015). *UML*. Obtido de Wikipédia, a enciclopédia livre: https://pt.wikipedia.org/wiki/UML